Facultad de Ingeniería

Ingeniería Mecatrónica

Programa Especial de Titulación

“Desarrollo de un sistema de reconocimiento de placas para la obtención de información en tiempo real, en apoyo a las operaciones de la empresa UNACEM, SAA, distrito de Villa María del Triunfo, provincia y departamento de Lima-Perú”

Mario Rodrigo Puente Lara

Para optar el Título Profesional de Ingeniero

Mecatrónico

Lima - Perú

2021

DEDICATORIA

A mi madre Ana Lara y a mi padre Mario Puente que me apoyaron incondicionalmente y me enseñaron valores para ser una buena persona. A mi hermano Juan Luis que está en mi corazón por siempre y a mi hermano Diego que me incentivo a ser un ejemplo a seguir.

AGRADECIMIENTO

Agradezco de todo corazón a mis padres, hermanos, abuelos y a toda mi familia por haberme apoyado en el transcurso de la vida, de igual manera agradezco a mi pareja por haber estado conmigo en los malos y buenos momentos, a mis profesores de toda la vida que me dieron el conocimiento y motivación para ser un profesional, a los amigos con los que pasamos momentos inolvidables y a los ingenieros/ técnicos de la fábrica Unacem donde aprendí a desenvolverme como profesional.

RESUMEN

El propósito del proyecto es obtener información en tiempo real de una máquina cuando el trabajador se encuentra en planta. En el proyecto se desarrolla un sistema de reconocimiento de placas que identifica el código de una máquina para obtener información de la placa eléctrica, placa mecánica, función en el proceso del cemento, las características y su plan de mantenimiento.

La metodología que se utiliza es el procesamiento de una imagen que consiste en convertir la imagen a escala de grises, transformarlo a una imagen binaria, filtrarlo para eliminar el ruido, utilizar operaciones morfológicas, detectar los bordes, detectar la placa y descifrar el código con el sistema de reconocimiento óptico de caracteres (OCR). Además, se realiza una interfaz gráfica que muestra la placa detectada junto a la información requerida.

Los resultados que se obtiene son eficientes siempre y cuando la captura de la imagen se da de manera frontal o cercana a esa posición a un metro de distancia dando como valor el código detectado con una exactitud de 95%.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCION

En la empresa Unacem los trabajadores del área de mantenimiento necesitan estar en planta para observar el problema en una máquina, reparar o sustituir un componente o realizar el mantenimiento semanal. La mayoría de veces los trabajadores necesitan información de alguna máquina solicitándolo a un trabajador de oficina ocasionando pérdidas de tiempo. Además, los nuevos trabajadores o practicantes necesitan aprender del proceso del cemento y al obtener información en tiempo real hace eficiente el proceso de aprendizaje. Internacionalmente varios países utilizan la tecnología Near Field Communication (NFC) para el ahorro de tiempo al acceso de información. Igualmente, a nivel Nacional se desarrolló una tesis del autor Herrera J (2013) con el propósito de agilizar el acceso de información de la pieza de un museo, en el cual se implementa un aplicativo móvil basado en la tecnología NFC, en el presente proyecto no se implementará esta tecnología, debido que se necesitaría poner a cada máquina un código que descifre la tecnología NFC en cambio se utiliza la visión artificial que aprovecha los códigos identificativos de cada máquina

En el presente proyecto se va a desarrollar mediante la tecnología de Visión artificial un sistema de reconocimiento de placas que agilice el acceso de información a los trabajadores.

En el capítulo 1, se presenta el problema que tiene la empresa, una solución, objetivos que se alcanzan, el alcance, la limitación del proyecto y el estado del arte. En el capítulo 2, se presenta el marco teórico que sirve de sustento para el desarrollo, además de los pasos que necesitamos seguir para el desarrollo del proyecto. En el capítulo 3, se desarrolla la solución del problema siguiendo los pasos de la metodología, además se encuentra el algoritmo de procesamiento de imágenes y la información que se brinda al trabajador. En el capítulo 4, se demuestra los resultados obtenidos por el sistema de reconocimiento.

CAPITULO 1

ASPECTOS GENERALES

1.1 Descripción del Problema

En la empresa Unacem los trabajadores que se encuentran en planta muchas veces solicitan información al personal que maneja el programa SAP, para conocer la placa eléctrica, mecánica o alguna característica, ocasionando tiempos muertos al recibir información. Además, los nuevos trabajadores o practicantes necesitan conocer el proceso del cemento en la planta y al no tener información en tiempo real dificulta el proceso de aprendizaje.

CAUSAS	EFECTOS
Deterioro en placas eléctricas y mecánicas	Solicitar información a oficina ocasionando pérdida de tiempo en la respuesta
Información de códigos identificativos de las máquinas solamente en oficina	Demora en el proceso de aprendizaje de los nuevos trabajadores ocasionando baja productividad
Información del proceso y mantenimiento solamente en oficinas	Demora en el proceso de aprendizaje de los nuevos trabajadores ocasionando baja productividad

1.2 Formulación del Problema

Una vez que se sabe que necesitamos obtener información en tiempo real para evitar la pérdida de tiempo que genera recibir información, se formula la siguiente pregunta.

¿Es eficiente el desarrollo del sistema de reconocimiento de placas identificativas para reconocer el código de la placa y obtener información en tiempo real?

1.3 Definición de objetivos

1.3.1 Objetivo General

- Desarrollo de un sistema de reconocimiento de placas mediante la tecnología de visión artificial para obtener información en tiempo real de las máquinas de la fábrica Unacem.

1.3.2 Objetivo Especifico

- Mostrar la placa identificada e información en tiempo real en una interfaz fácil de interpretar
- Mostrar únicamente la información de las placas que se encuentran en los datos almacenados de forma correcta
- Identificar el procedimiento óptimo para obtener el código correcto de la placa identificativa.

1.4 Alcances y limitaciones

1.4.1 Alcances

El proyecto tiene como alcance implementar un sistema de visión artificial para obtener el código de la placa identificativa de una máquina. No se desarrollará el sistema de información compuesto por un aplicativo móvil.

1.4.2 Limitaciones

El operario debe lograr que el sistema reconozca la placa, y obtenga el código identificativo para tener acceso a toda la información de la máquina. Para ello el

operario debe de tomar la imagen de manera frontal para obtener un código correcto a un metro de distancia y debe de esperar como máximo dos segundos de procesamiento para que se obtenga el código correcto.

1.5 Justificación

Es necesario que todo trabajador tenga información en tiempo real respecto a las características técnicas, planes de mantenimiento, fechas de revisión, de una máquina. De esta manera el trabajo del operario sería más eficiente debido que se obtiene información en tiempo real, reduciendo el tiempo de espera del trabajador, por ello se desarrolla un sistema de reconocimiento de placas que brinde información a los trabajadores. Las justificaciones son tecnológicas, porque le da a una empresa la innovación de tener su propia aplicación con visión artificial; justificación social debido a que beneficia a los trabajadores al acceso rápido de información además que facilita al digitar los códigos escaneados por visión artificial

1.6 Estado del arte

1.6.1 Antecedentes nacionales

- El autor Gerardo Alfonso (2014) de la tesis, con título “Sistema De Reconocimiento De Patrones En Placas Vehiculares Para El Acceso Automático De Visitas A Un Edificio” El aporte que se obtiene de la tesis es el procesamiento que realiza en el reconocimiento de la placa con el Software Matlab, los pasos a seguir para reconocer una placa de un automóvil y las etapas de pre-procesamiento de una imagen, para finalmente obtener el código de la placa con el sistema de reconocimiento OCR, la finalidad del autor es automatizar el ingreso de personas autorizadas en un edificio con visión artificial que resulta de manera éxitosa y recomienda tener una buena iluminación, posicionamiento, perspectiva de la placa y que algún sticker o adorno puede entorpecer el procesamiento

- Los autores Robert Barreto y David Lizarraga (2019) de la tesis con título “Modelo de Sistema de Reconocimiento Facial para el Control de la Trata de Personas””. El aporte que se obtiene de la tesis es el procesamiento que realiza a una imagen a través del lenguaje Python utilizando la librería OpenCv que es fundamental debido a que es la librería que se utilizara en nuestro proyecto por la alta eficiencia y desarrollo en la visión artificial, los autores muestran el procesamiento de la imagen, la retroalimentación a las redes neuronales, la resolución de la cámara que influyen en el procesamiento, además de poder mandar datos cuando se reconoce una imagen deseada en la cual nos ayuda para poder desarrollar nuestro proyecto

- El autor Herrera J. (2013) de la tesis con título “Diseño e implementación de una aplicación móvil basada en la tecnología nfc para acceso a información de las piezas de arte de un museo”. El aporte que se obtiene de la tesis es el procedimiento que realiza para obtener información en tiempo real con un aplicativo móvil, además de la conexión Web- móvil y como transferir la base de datos. Estos no se realizarán en el proyecto, pero es esencial para darnos una idea de cómo detectar las placas identificativas en tiempo real con un aplicativo móvil. El autor muestra la eficiencia del proyecto a la hora de obtener información de las piezas en tiempo real.

1.6.2 Antecedentes internacionales

- El autor Pedro G. (2013) de la tesis con título “Reconocimiento de imágenes utilizando redes neuronales artificiales”. El aporte que se obtiene de la tesis es el procedimiento que realiza para poder procesar una imagen y detectar el contorno

deseado del objeto que se desea extraer alguna característica, en nuestro caso se desea detectar la placa identificativa del fondo y reconocer los caracteres de la placa, el autor recomienda siempre tener una buena la iluminación y detectar la imagen de manera frontal para que funcione sus etapas de procesamiento.

- Los autores Juan Paredes, Leidy Guerrero (2012) de Ecuador con nombre de la tésis “Estudio comparativo entre algoritmos de reconocimiento de borde para identificación de placas de autos” El aporte que se obtiene de la tésis es la elección eficiente entre los algoritmos de reconocimientos de bordes para una placa, entre los algoritmos se encuentra el Sobel y el Canny, que están en las librerías de Open CV. Resultando que el algoritmo Canny es mejor para detectar un borde debido a su adaptabilidad y de no disminuir su performance ante la presencia de ruido, debido a que el Sobel elimina un gran porcentaje del ruido, pero se pierden regiones de interés de una placa provocando dificultad en la detección

- El autor Javier Pérez (2014) de Ecuador con nombre de la tésis “Reconocimiento de placas vehiculares mediante procesamiento de imágenes para optimizar el acceso a los parqueaderos de la UTA, Campus Huachi”

  El aporte que se obtiene de la tésis es el procesamiento que utiliza mediante el software Labview, muy diferente a los anteriores antecedentes, dando una mejor visíon a la hora de procesar una imagen debido a su implementación del proyecto y mostrando en tiempo real la detección de la placa, además demuestra una comunicación fiable entre el circuito, ordenador y el panel de control, recomendando que las letras de las placas sean del mismo estilo y entendibles para un mejor reconocimiento

CAPÍTULO 2

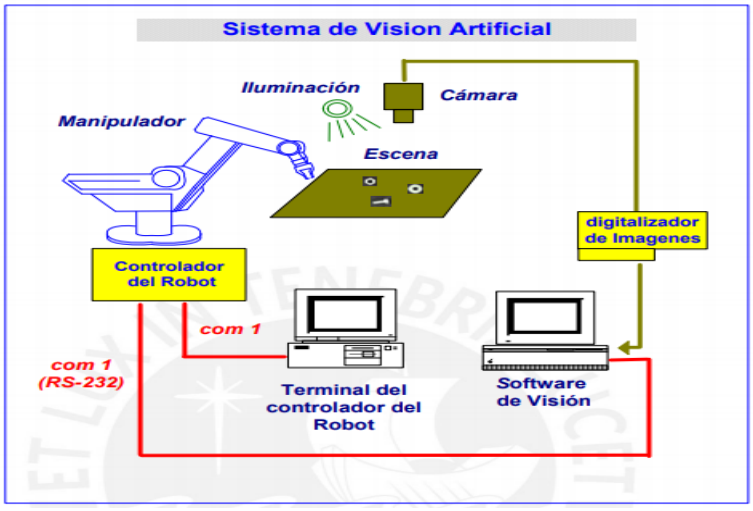
MARCO TEÓRICO

2.1 Fundamento teórico

2.1.1 Visión Artificial

La visión artificial son métodos para poder adquirir, procesar, analizar una imagen ya sea convirtiendo una imagen real en información numérica o simbólica para que sea procesado en una computadora, tal cual nosotros como personas procesamos una imagen con la ayuda de nuestros ojos y el cerebro la computadora necesita una cámara y un procesador. Es una parte de la inteligencia artificial en el cual nos permite crear algoritmos, aplicaciones para obtener información espacial (3D) y procesar varias imágenes digitales (2D) [17]

Figura 1:Sistema de Visión Artificial



Fuente: Sobrado, E (2003)

2.1.1.1 Elementos de los sistemas de Visión artificial

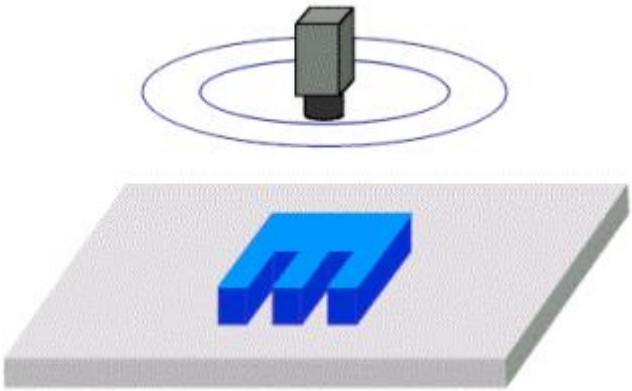
2.1.1.2 Sistema de iluminación

La iluminación es esencial en el desarrollo de los sistemas de visión artificial para obtener resultados óptimos. Teniendo una iluminación adecuada se obtiene una mayor exactitud en las medidas. Los principales objetivos de la iluminación en los sistemas de visión artificial son: mantener la intensidad constante y tener un buen contraste para diferenciar los objetos del fondo [17]. Por tanto, se verá el tipo de iluminación más utilizado

2.1.1.3 Iluminación Frontal

Es el tipo de iluminación más utilizado y está encargado de iluminar de manera frontal a la pieza. Uno de los problemas que presenta es la obtención de un buen contraste en la pieza y el fondo, realizado por sombras y brillos que alteran al procesamiento del objeto [16]

Figura 2: Iluminación de manera Frontal



Fuente: Sobrado, E (2003)

2.1.1.4 Cámara digital

Una cámara digital captura y almacena las imágenes de manera digital para que pueda ser procesada por una computadora**,** la creación de una cámara digital se origina con el

objetivo de dar información de navegación a los astronautas a bordo de misiones espaciales. Anteriormente las cámaras digitales capturaban imágenes para convertirlas en señales eléctricas y a su vez almacenarlos en soportes magnéticos. Pero en 1988 la cámara DS-1P pudo almacenar imágenes en una computadora. [18]

En la actualidad existen diversas cámaras digitales que tienen la función de grabar sonido y videos además de capturar imágenes e incluso existen cámaras incorporadas en otros dispositivos como en el caso de los celulares

Figura 3 Cámara digital

Fuente: Nieto, A (2018)

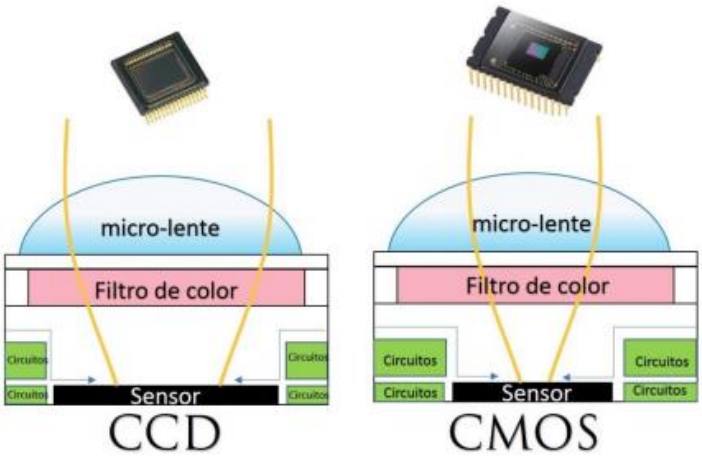
2.1.1.5 Sensores de una cámara digital

Generalmente las cámaras digitales tienen 2 tipos de sensores:

Sensor CCD (Dispositivo de carga acoplada), es un dispositivo que tiene ventajas en el tamaño y la calidad de la imagen, debido que reduce el ruido y disminuye la sensibilidad según sea mayor la resolución

Sensor CMOS (Semiconductor complementario de óxido metálico), Se reconoce con ese nombre debido al tipo de tecnología que utiliza, este sensor ofrece buenos resultados aún con poca luz, pero al ser menos sensible a la luz ofrece menos calidad en una imagen y la posibilidad de tener efecto de ruido. [17]

Figura 4: Sensores de una cámara digital



Fuente: Nieto, A (2018)

2.1.1.6 Módulo de proceso

El módulo del proceso puede ser una computadora o un sistema integrado en el cual se almacenan las imágenes y se procesa por algoritmos que pueden extraer información necesaria y tomar decisiones, según la aplicación del sistema de visión artificial. [20]

Figura 5: Sistema integrado Raspberry Pi

Fuente: Castillo, J (2020)

2.1.2 Representación y codificación de una imagen digital

2.1.2.1 Pixel

El pixel se encuentra como dato en una matriz de 2 dimensiones de una imagen digital en la cual es la unidad de una imagen que se encuentra en imágenes compuestas por un mapa de bits. El pixel tiene tres características [20]

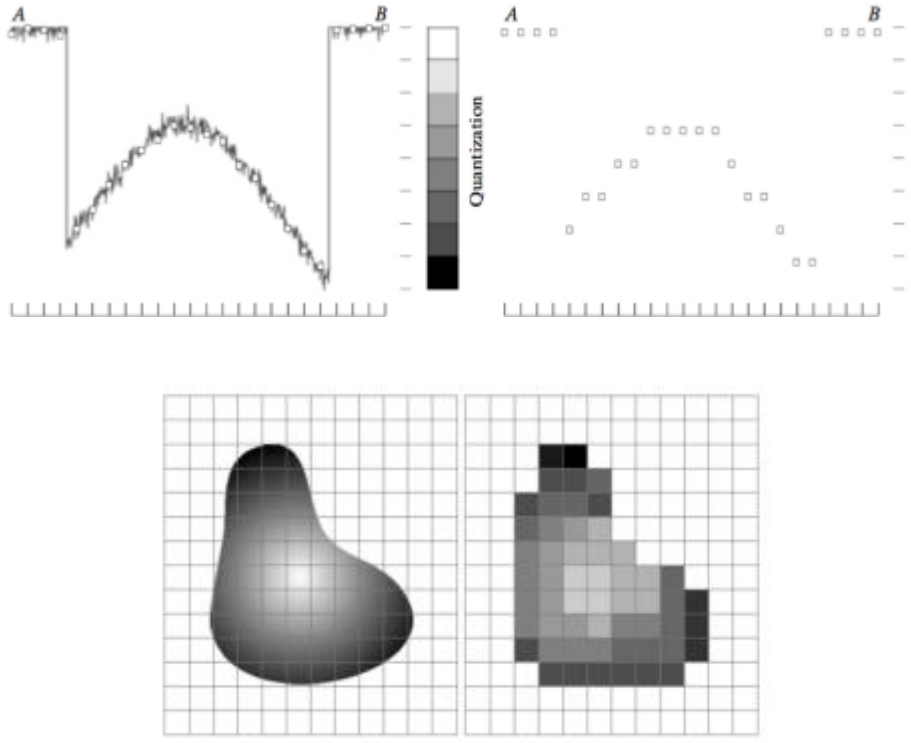
- Forma Cuadrada
- Capaz de almacenar color
- Posición relativa a los demás pixeles en un mapa de bits

2.1.2.2 Muestreo y Cuantificación

Se debe de pasar por dos procesos para obtener una imagen digitalizada. El primero es el Proceso de Muestreo que consiste en convertir una señal continua en una secuencia

numérica (Discreta), el teorema de muestreo es comúnmente llamado teorema de muestreo de Shannon, y el Segundo proceso es el de cuantificación que consiste en digitalizar la amplitud de los valores [19]

Figura 6: Muestreo y cuantificación de una imagen



Fuente: Boris, R (2006)

2.1.2.3 Resolución

Es determinado por los pixeles de ancho por los de alto que especifican un área rectangular donde se encuentra la imagen. Algunas de las resoluciones más comunes son [20]

- 640x480
- 800x600
- 1024x768

2.1.2.4 Profundidad de color

La profundidad de color informa la cantidad de colores que puede contener cada pixel que conforma una imagen. Mientras más bits de información haya por pixel entonces se encontrará disponible más colores y la representación de la imagen será más exacta, por ejemplo, para los colores indexados RGB, si hubiera 1 bit por pixel entonces habría 2 colores disponibles, pero si hubiera 2 bits por pixel hubiera 4 colores y si hay 3 bits por pixel habría 8 colores disponibles, así sucesivamente se consigue la cantidad de colores disponibles siguiendo la secuencia de 2 elevado a la cantidad de bits [20]

2.1.2.5 Contraste

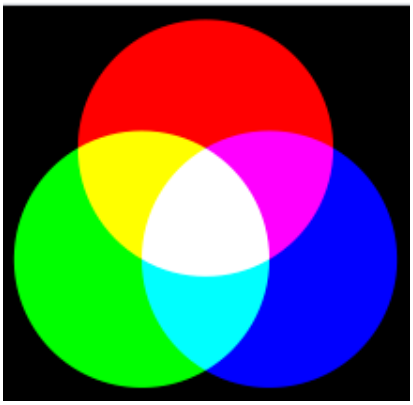
El contraste es la diferencia entre el tono más oscuro y el más blanco de una imagen, se puede decir que una imagen tiene mucho contraste cuando la tonalidad de la imagen esta esparcida por todo el rango de tonos [20]

2.1.2.6 Clasificación de imágenes digitales

2.1.2.7 Imagen a colores RGB

Es un modelo basado en conformar mediante una combinación de 3 colores primarios ya sean el rojo, verde y azul que contienen en total 256 tonos o brillos, por lo tanto, con la fórmula de 2 elevado a la cantidad de bits se obtiene que se necesita 8 bits por cada color ya sea rojo, verde o azul, sumando las 3 cantidades en total se necesitan 3 bytes por cada pixel, es decir 24 bits [20]

Figura 7: Colores RGB



Fuente: López, J (2009)

2.1.2.8 Imagen en escala de grises

Es una escala empleada en las imágenes digitales para representar las intensidades. En el cual el numero cero equivale al nivel más oscuro (Negro) y el nivel más alto es el tono más brillante (Blanco) [20]

2.1.2.9 Imagen binaria

Es una imagen digital representada en colores blancos 0 y negros 1, anteriormente se realiza una umbralización para poder descifrar de que tono a que tono la imagen en escala de gris se pasa a binaria [20]

2.1.3 Lenguajes de programación

Actualmente existen varios lenguajes de programación como Matlab, C++, y en nuestro caso desarrollamos con el lenguaje Python, Los lenguajes mencionados son los que más se utilizan en la aplicación de la industria de Visión artificial.

2.1.3.1 Matlab

Matlab es un entorno de software matemático que en un principio se creó para tener acceso al software de matriz desarrollado por proyectos LINPACK y EISPACK. Utiliza un lenguaje de programación propio (Lenguaje M) y está disponible para las plataformas UNIX, Windows, Mac OS X. [4]

2.1.3.2 Python

Es un lenguaje de programación multiparadigma ya que soporta orientación a objetos, es lenguaje interpretado dinámico y de multiplataforma, es de escritura rápido, escalable robusta y de código abierto, ventajas que hacen que Python sea perfecto para inteligencia artificial. Permite desarrollar ideas complejas en pocas líneas de código lo que no es posible en otros lenguajes [8]

2.1.4 Librerías

2.1.4.1 OpenCV

OpenCV (Open Source Computer Vision) es una biblioteca libre de visión artificial originalmente desarrollada por INTEL Es la más utilizada actualmente para el procesamiento de imágenes además de ser utilizados en control de procesos, detección de movimiento, reconocimiento objetos, reconocimientos de rostros, robótica, etc [10]

2.1.4.2 Tesseract

Es un motor de reconocimiento de caracteres disponible en varios sistemas operativos siendo un software libre, es de código abierto y es uno de los tres motores con mayor precisión a la hora de detectar los caracteres [28]

2.1.4.3 Tkinter

Es una librería que facilita el desarrollo de una interfaz gráfica. Con esta librería se puede crear ventanas emergentes, visualizar variables en tiempo real, graficar datos, en este proyecto lo utilizaremos para tener una interfaz además de poder capturar el código con un botón predefinido [29]

2.1.5 Procesamiento de imágenes

Para poder realizar el procesamiento de una imagen se necesita conocer a detalle su contenido.

2.1.5.1 Adquisición de imágenes

Es la etapa más importante para procesar una imagen debido a que toma cientos de muestras por segundo para que pueda ser procesado luego en el computador.

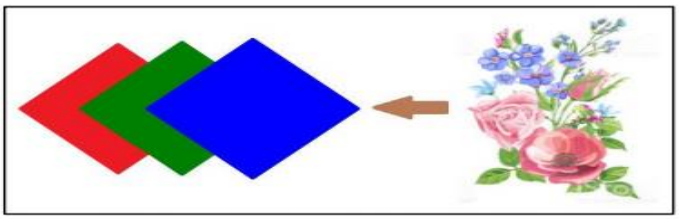
Según el lenguaje Python utilizando OpenCv la adquisición de la imagen se da a través de una cámara con el siguiente código: [1]

```
Captura=cv2.Videocapture('Imagen.mp4')
```

2.1.5.2 Conversión a escala de grises

Toda imagen digital se compone de tres planos de tonos primarios, Rojo Verde y azul, en la cual realiza un mesclado de los tres canales RGB para obtener un solo gris tomando ciertos porcentajes como Rojo 30%, Verde 59% y azul 11%. En el cual es lo más parecido a lo que los ojos humanos pueden captar la intensidad de la luz [1]

Figura 8: Planos de los colores RGB



Fuente: Castillo, J (2020)

Según el lenguaje Python y usando OpenCv, la conversión a escala de grises se da con el siguiente código

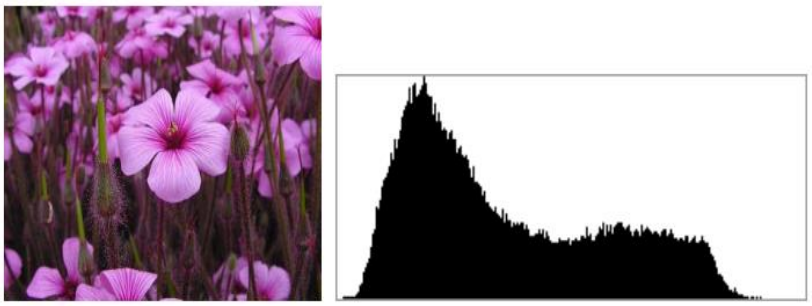
```
Gray=cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
```

2.1.5.3 Histograma

El histograma representa de manera gráfica las frecuencias que pertenecen a los distintos colores en una imagen, nos proporciona información valiosa para juzgar las características de luminosidad de una imagen como el brillo y el contraste. El histograma se puede representar a través de una función que muestra la cantidad de pixeles que tiene un nivel de gris respectivo. [21]

Se puede observar en las siguientes figuras con 256 niveles de brillo, como se utiliza el histograma para poder diferenciar la cantidad de tonalidad que se encuentra en cada figura

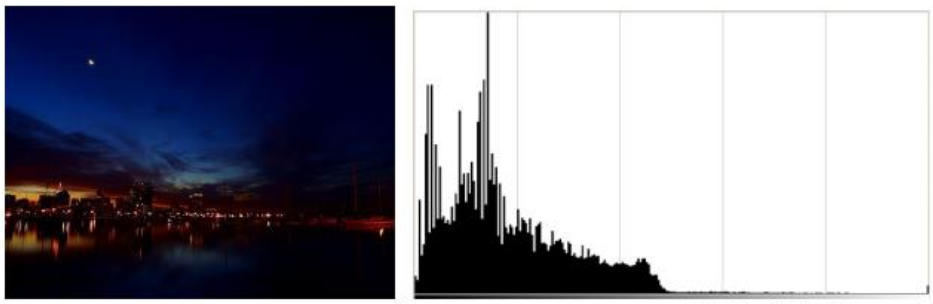
Figura 9: Histograma que representa un buen contraste



Fuente: Vicente, V (2015)

En la siguiente figura 10 se puede observar como el histograma se encuentra desplazado en el lado izquierdo que pertenece a la zona oscura y evidenciando que la zona brillante es muy pequeña, debido que es una imagen nocturna

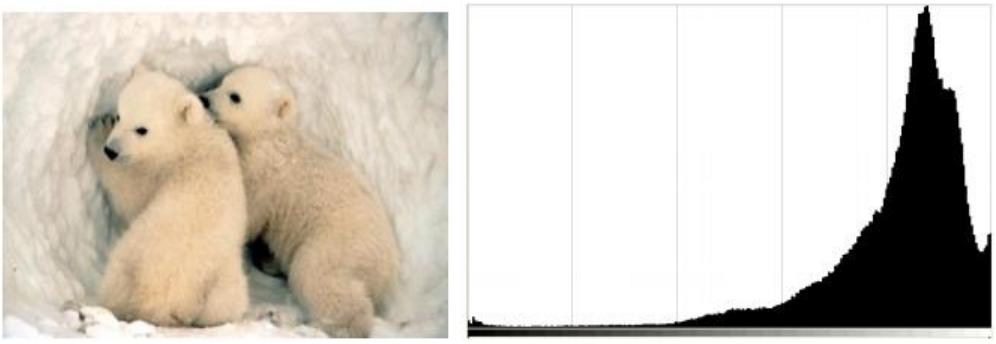
Figura 10: Histograma que representa una imagen oscura



Fuente: Vicente, V (2015)

Como otro ejemplo se muestra cómo se comporta el histograma en una imagen que tiene bastante claridad, se observa que el histograma está en el lado derecho donde pertenece a la zona más alta de grises claros y blancos

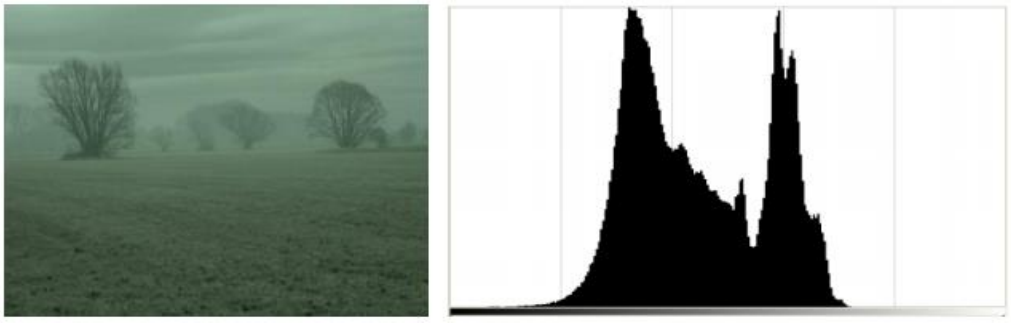
Figura 11: Histograma que representa una imagen clara



Fuente: Vicente, V (2015)

Como último ejemplo se muestra una imagen que contiene un bajo contraste, debido a la pequeña diferencia de brillo entre las zonas claras y oscuras, además se observa que no hay nada completamente oscuro ni claro y como la tónica que domina son los grises medios se observa un histograma centrado

Figura 12: Histograma que representa a una imagen con bajo contraste



Fuente: Vicente, V (2015)

2.1.5.4 Umbralización

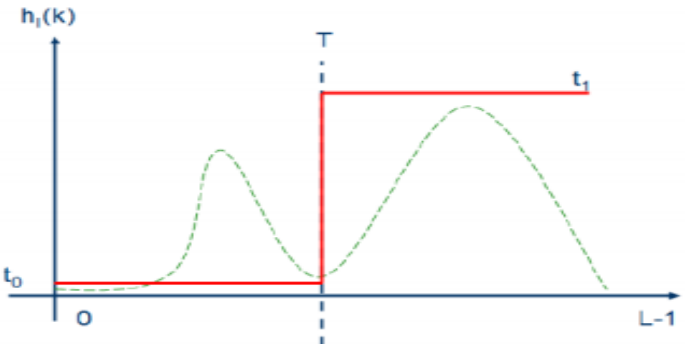
Para poder realizar la binarización se necesita obtener un valor de pixel adecuado llamado valor umbral, en el cual se segmenta de manera simple y eficiente separando los pixeles

de una imagen en escala de grises a partir de ese valor. Esta se obtiene observando el histograma de la imagen y se puede realizar a través de 2 maneras [22]

2.1.5.4.1 Umbralización global

El umbral global o fijo es único en toda la imagen, en la cual se obtiene observando el histograma y eligiendo de diferentes métodos el valor del umbral. Esta estrategia de umbralización global resulta útil cuando la iluminación es homogénea sobre la imagen y tiene un histograma bimodal (Se puede diferenciar entre la parte oscura que la clara) [22]

Figura 13: Umbral seleccionado según histograma bimodal



Fuente: Triana,N., Jaramillo, A., Gutierrez R,. Cesar R (2016)

La función se vuelve binario si

$$b(r,c) = 0 \text{ Si } I(r,c) \leq T \quad y \quad b(r,c) = 1 \text{ Si } I(r,c) > T$$

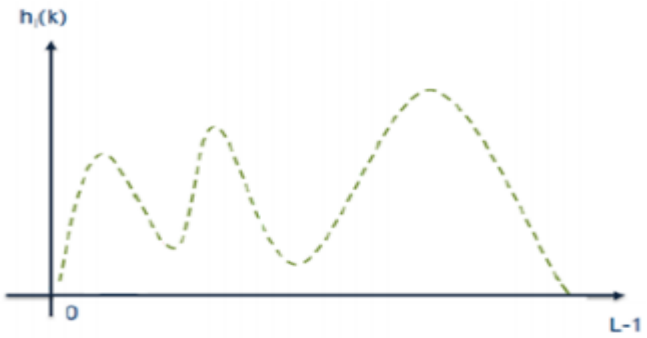
Donde I (r, c) Es un valor del pixel (r, c) de la imagen de nivel de gris, b (r, c) es la imagen binaria y T es el valor del Umbral

2.1.5.4.2 Umbralización local o adaptativa

Mayormente cuando una imagen presenta una distribución no Bimodal (que no se puede diferenciar entre los puntos más oscuros que los claros) debido a los cambios bruscos de

iluminación, se divide la imagen en subimágenes con el objetivo de encontrar un umbral para cada una de ellas, las subimágenes pueden ser de diferentes formas y tamaños por lo que se mantiene definido antes de implementar el algoritmo, de esta manera no existe un único umbral para toda la imagen, sino múltiples umbrales para cada sub imagen. [22]

Figura 14: Umbral seleccionado según histograma no bimodal



Fuente: Triana,N., Jaramillo, A., Gutierrez R,. Cesar R (2016)

2.1.5.4.3 Métodos de cálculo del Umbral

Para obtener el Valor del umbral se realiza de diferentes maneras en el cual los tipos son, el promedio de máximo y mínimo, diferencia entre máximo y mínimo y mínimo local [22]

2.1.5.4.3.1 Promedio de máximo y mínimo

Determina el valor medio que hay en la intensidad máxima y mínima del histograma, se representa según la siguiente ecuación [22]

$$\mathrm{T} = (\max + \min)/2$$

2.1.5.4.3.2 Diferencia entre máximo y mínimo

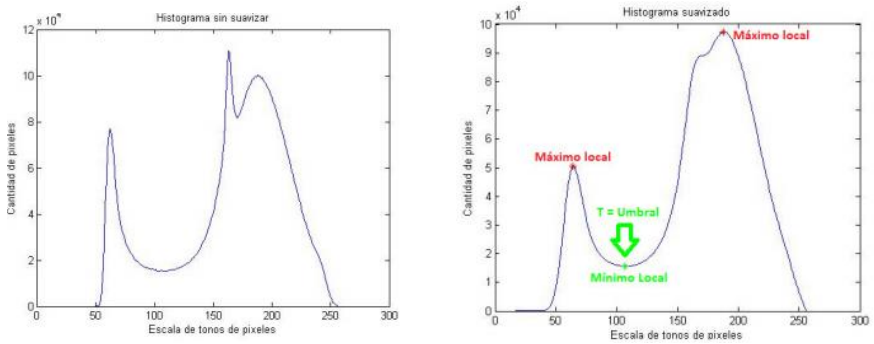
Es similar al promedio máximo y mínimo, pero en este caso se obtiene el valor medio entre la diferencia de la Intensidad máxima y mínima del histograma [22]

$$\mathrm{T} = (\max - \min)/2$$

2.1.5.4.3.3 Mínimo local

Se observa el histograma de manera visual y se calcula el valor mínimo entre los picos del histograma, normalmente para realizar ese cálculo se realiza el suavizado de la curva [22]

Figura 15: Mínimo local del histograma



Fuente: Triana,N., Jaramillo, A., Gutierrez R,. Cesar R (2016)

2.1.5.5 Transformación de una imagen a binaria

Es la imagen que se puede diferenciar entre el blanco y el negro dependiendo la umbralizacion que se escoja. Según la umbralización se selecciona de que valor a que valor se pone en blanco o en negro, en el siguiente código de OpenCv se observa que de 0 a 200 es blanco y luego es negro [1]

```
_, Binario = cv2.threshold(image, 200, 255, cv2.THRESH_BINARY).
```

2.1.5.6 Ruido

Toda imagen tiene una cierta cantidad de ruido el cual se debe a la cámara o el medio de transmisión de la señal, el ruido se manifiesta como pixeles aislados que toman un nivel gris diferente al de sus vecinos. El algoritmo del filtrado te permite eliminar o disminuir el

ruido, existen varios tipos de ruidos, alguno de ellos son el Gaussiano y el sal y pimienta (Impulso) [1]

2.1.5.6.1 Gaussiano

Produce pequeñas variaciones en una imagen, debido generalmente por ganancias en la cámara IP, ruido en los digitalizadores, perturbaciones en la transmisión, etc. [1]

2.1.5.6.2 Sal y pimienta

El valor que se toma en el pixel no tiene relación con el valor ideal, sino con el valor del ruido que toma valores muy altos y bajos [1]

2.1.5.7 Filtrado de una imagen

El filtrado es una técnica que sirve para mejorar una imagen, en el cual se le dice una operación de vecindario porque el valor del pixel dado en la imagen procesada se calcula mediante un algoritmo que toma en cuenta los valores de los pixeles vecinos de la imagen original, existen varios tipos de filtros digitales ya sea para eliminar altas o bajas frecuencias, suavizar la imagen, realzar o detectar bordes, además sirven para eliminar los distintos ruidos ya mencionados anteriormente, el proceso de filtrado se lleva a cabo sobre el dominio del espacio y en el dominio de la frecuencia [23]

2.1.5.7.1 Filtros en el dominio del espacio

Tiene como función la modificación de rangos de frecuencias en una imagen, el termino espacial indica que el filtro es aplicado de manera directa a la imagen y no por una

transformada, el nivel de gris se obtiene en función al valor de sus vecinos, se clasifican en filtros lineales y no lineales [23]

2.1.5.7.1.1 Filtros lineales

Las operaciones lineales se conforman en la multiplicación de cada pixel en el vecindario por un coeficiente respectivo y realizar la suma al resultado, existen dos conceptos cuando se realiza un filtrado espaciado lineal, uno es correlación y el otro es convolución. La correlación consiste en pasar la máscara por la imagen mientras que la convolución, la máscara se pasa por la imagen, pero antes se rota 180 grados. [23]

En la siguiente ecuación se muestra como ejemplo una imagen I(k,I) y una máscara h(k,I), dando como resultado una imagen O(x,y) [20]

$$O(x,y) = \sum_{(k,l)\in E_N} \sum h(k,l) I(x-k, y-l)$$

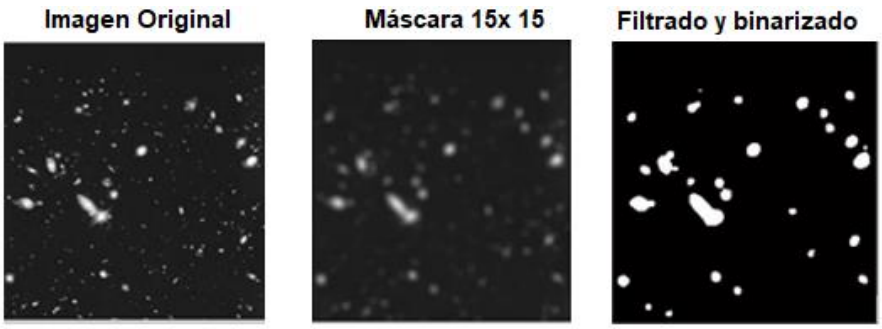
Existen filtros que pertenecen a este tipo de clase como los siguientes

2.1.5.7.1.1.1 Filtro de la media

En el filtro de la media se reemplaza el valor de cada pixel por la media de los valores de los pixeles vecinos, la ventaja que tiene este filtro es que es simple intuitivo y fácil de implementar.

Las aplicaciones más comunes que se utiliza este filtro son para suavizar las imágenes, detector los objetos de mayor tamaño y eliminar el ruido [24]

Figura 16: Imagen filtrado con un filtro tipo media



Fuente: R.C. González, R.E. Woods (2008)

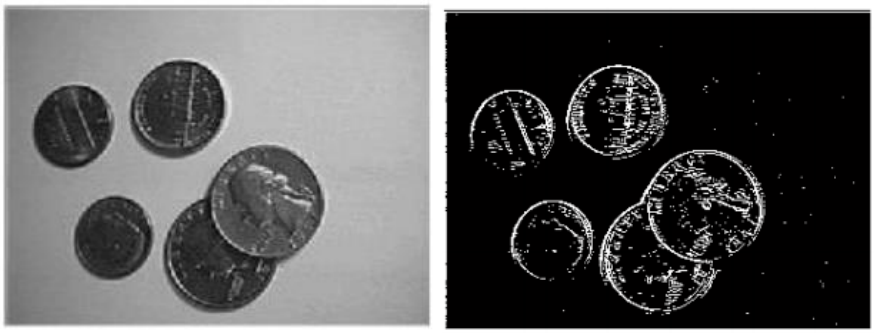
2.1.5.7.1.1.2 Filtros de Realce:

El objetivo de un filtro de realce es mostrar los detalles de una imagen que se pueden haber borrado, este filtro está relacionado con la detección de los bordes en el cual es el cálculo de un operador local de derivación debido que un pixel pertenece a un borde cuando sufre un cambio brusco entre niveles grises con sus vecinos, si el cambio es más brusco, el borde es detectado fácilmente, en este grupo de realces se encuentras los operadores Robert, Laplaciano y Sobel. Se va a observar los operadores como ejemplos de Realce [24]

2.1.5.7.1.1.2.1 Operador Roberts

Se usan dos máscaras, las ventajas es que es fácil y rápido de computarizar ya que solo implica un entorno de 4 pixeles y solo se realiza sumas y restas, una de las desventajas es que es sensible al ruido y tiene respuesta débil a los verdaderos bordes [24]

Figura 17: Imagen filtrado con un tipo Roberts

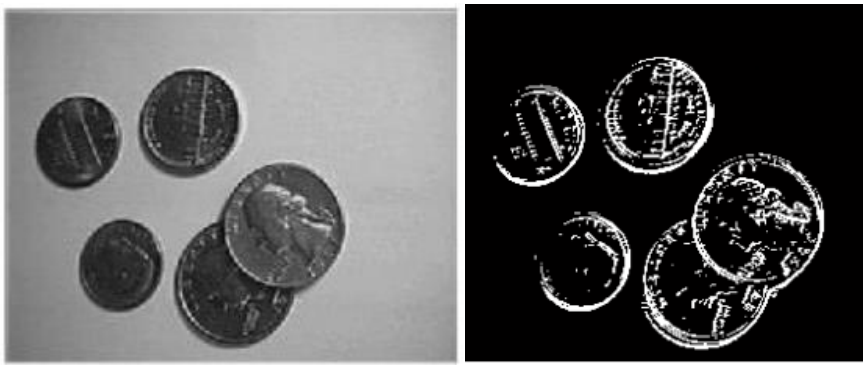


Fuente: R.C. González, R.E. Woods (2008)

2.1.5.7.1.1.2.2 Operador Sobel

Utiliza dos máscaras de 3x3, la ventaja respecto a Sobel es que es menos sensible al ruido pero la desventaja respecto a Sobel es que mas lento al procesarse [24]

Figura 18: Imagen filtrado con un tipo Sobel



Fuente: R.C. González, R.E. Woods (2008)

2.1.5.7.1.1.2.3 Operador Laplaciano

No suele utilizarse directamente en las prácticas debido a que es muy sensible al ruido, pero se utilizan sumando o restando según la máscara con la imagen original para realzar los contornos [24]

Figura 19: Imagen filtrado con un operador Laplaciano

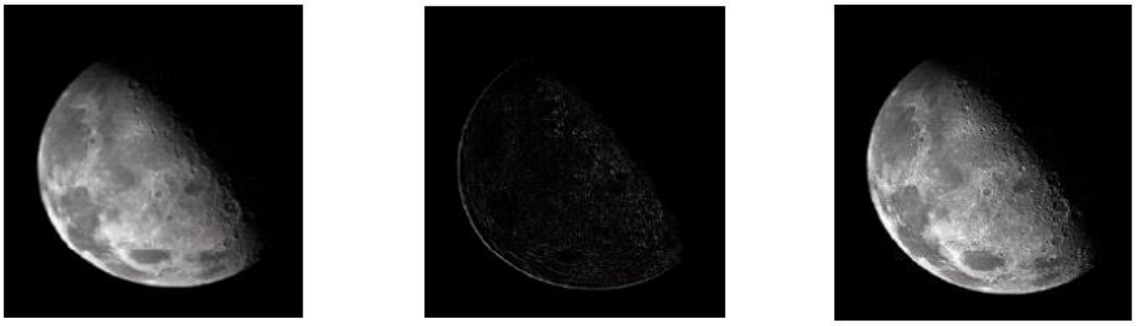


Imagen Mascara Resultado

Fuente: R.C. González, R.E. Woods (2008)

2.1.5.7.1.2 Filtros no lineales

El filtrado no lineal se basa en operaciones de vecindario al igual que en los filtros lineales, pero mientras que el espaciado lineal se basa en calcular la suma de los productos en una operación lineal, el filtrado no lineal se basa en operaciones no lineales sobre pixeles de un vecindario. [24]

Se puede representar según la siguiente ecuación, en el cual h(I,j) es la máscara, f(I,j) la imagen original y g(I,j) es el resultado de la operación [20]

$$g(i,j) = O_{m,n}[h(m,n)f(i-m,j-n)]$$

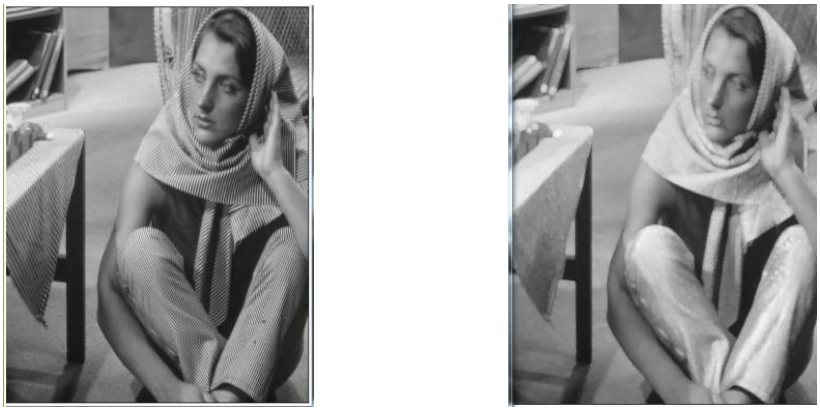
Existen filtros que pertenecen a este tipo de clase como los siguientes.

2.1.5.7.1.2.1 Filtros de estadísticos ordenados

El filtro se encarga de ordenar los valores en la vecindad de cada pixel de menor a mayor y obtiene un valor de la lista respectiva, existen diferentes tipos como: [24]

- Filtro máximo: Hace una selección de la lista ordenada de valores de nivel gris, tomando como referencia al mayor valor, la ventaja es que elimina el ruido pimienta, pero las desventajas son que solo funciona cuando el ruido es únicamente pimienta y tiende a aclarar la imagen dificultando el procesamiento [24]

Figura 20: Imagen filtrado con un filtro máximo



Fuente: R.C. González, R.E. Woods (2008)

- Filtro Mínimo: Hace una selección de la lista ordenada de valores de pixeles tomando como referencia al menor valor, la ventaja es que elimina el ruido sal, pero la desventaja es que solamente funciona para el ruido sal y tiende a oscurecer la imagen, dificultando de igual manera el procesamiento [24]

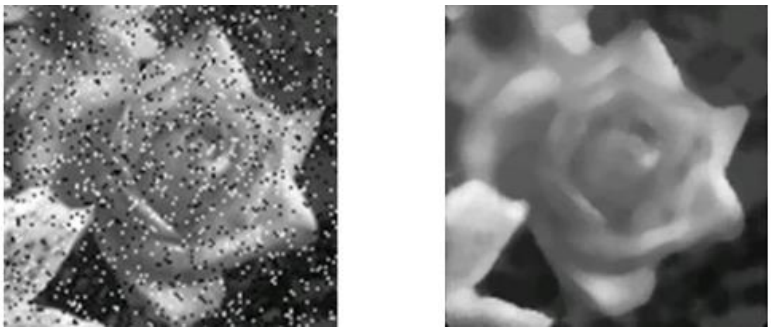
Figura 21: Imagen filtrado con un filtro mínimo

Fuente: R.C. González, R.E. Woods (2008)

- Filtro de la mediana: Hace una selección de la lista ordenada de valores de pixeles tomando como referencia el valor en la posición intermedia (mediana), las ventajas que tiene este filtro es que atenúa el ruido sal y pimiento, elimina los efectos engañosos y preserva el borde de la imagen, pero las desventajas son que no es lineal, pierde detalles (puntos, lineas finas) y redondea las esquinas de los objetos en el cual puede ocasionar problemas en el procesamiento

  Existen otros tipos de filtros en los cuales algunos son la mediana ponderada, punto medio del entorno de vecindad y media geométrica. [24]

Figura 22: Imagen filtrado con un filtro mediana



Fuente: R.C. González, R.E. Woods (2008)

2.1.5.7.2 Filtros en el dominio de la frecuencia

Son utilizados para eliminar las altas o bajas frecuencias de una imagen, con estos filtros se puede logra suavizar, realzar o detectar bordes. Estos filtros trabajan con la transformada discreta de Fourier [20]. Principalmente existen 3 tipos:

- Filtro paso bajo: Consiste en eliminar o atenuar las altas frecuencias de una imagen para quedarse con las bajas frecuencias. Para lograr el objetivo se necesita una máscara binaria de manera que cuando se aplica sobre el espectro de frecuencias mantiene los valores bajos que se sitúan en el centro de la imagen y elimina los valores altos. [20]
- -Filtros pasa alto: Consiste en eliminar o atenuar las bajas frecuencias de una imagen para quedarse con las altas frecuencias. Para lograr el objetivo se necesita una máscara circular tal que dentro del círculo tenga como valor cero y fuera del mismo la unidad. Mientras más grande sea el radio del círculo, el intervalo que se elimina de bajas frecuencias será mayor y mejor será el efecto de filtrado [20]

- Filtros de paso de banda: Consiste en eliminar las frecuencias intermedias o extremas de una imagen para quedarse con las frecuencias que están en una banda determinada, para lograr el objetivo se necesita una máscara binaria en forma de anillo con un radio interno y otro externo. El opuesto a este filtro es el rechazo de banda que consiste en eliminar las frecuencias de la banda para mantener las frecuencias fuera de ella [20]

2.1.5.8 Transformada rápida de Fourier

Es un eficiente algoritmo en el cual tiene la ventaja de ser la version computarizada eficiente de la transformada discreta (DFT) debido a su ahorro en las operaciones matemáticas, permite calcular el DFT y la inversa. Es aplicado en el tratamiento digital de señales y el filtrado digital. [25]

La DFT es definida por la siguiente ecuación

$$F(\mu) = \frac{1}{N} \sum_{x=0}^{N-1} f(x).e^{-j\,2\pi\mu x/N}$$

Donde la formula requiere de N^2operaciones artiméticas, pero mediante un algoritmo FFT se obtiene el mismo resultado reduciendo las operaciones en solo $(\mathrm{N}.log_2(\mathrm{N}))$ [25]

2.1.5.9 Algoritmo Canny

Es un algoritmo para poder detectar bordes, desarrollado por John Canny en el cual se reduce el ruido. Se encuentra el gradiente de intensidad de imagen, se hace una supresión no máxima y se encuentra el umbral de histéresis para poder decidir cuáles son los bordes y cuáles no. Para poder hallar la detección de los bordes se realiza el siguiente código [10]

```
Bordes=cv2.Canny (img,100,200)
```

2.1.5.10 Operación morfológica

2.1.5.10.1 Dilatación

Sirve para poder engrosar las líneas blancas de una imagen, cambiando el tamaño de grosor y esto se puede variar con el siguiente código [1]

```
Dilate=cv2.dilate(img, kernel, iterations=1)
```

Figura 23: Operación dilatación en una imagen

Fuente: Castillo, J (2020)

2.1.5.10.2 Erosión

Sirve para hacer las delgado las líneas blancas de una imagen, cambiando el tamaño del grosor y esto se varía según el código [1]

```
Erosion=cv2.erode(img, kernel, iterations=1)
```

Figura 24: Operación erosión en una imagen

Fuente: Castillo, J (2020)

2.1.5.11 Extracción de características

Encargado de observar todas las características que tiene un objeto para poder discriminar a los objetos que no cumplen con ciertas características [27], las principales características que se obtienen son:

- Área
- Perímetro
- Centroide
- Área Convexa
- Excentricidad

En OpenCv2 se crea un vector que contiene a los contornos en una imagen con el siguiente código findContours

2.1.5.12 Clasificación (Discriminación de contornos)

Esta etapa se encarga de poner limitaciones a las características que debe de tener el objeto deseado, por ejemplo, un cierto grano de cacao debe de contener un área mínimo de 2500 y una altura de 67.30, pero si al procesar un cacao obtiene un área menor a 2500 y altura menor a 67.30 entonces en conclusión programa no detectará como un cacao.[27]

2.1.5.13 Detección de caracteres

La detección de caracteres se puede dar de diferentes maneras como el KNN, por redes neuronales específicos o por el OCR Tesseract.

2.1.5.13.1 Reconocimiento mediante KNN.

Es un algoritmo de aprendizaje supervisado, que necesita ser entrenado con una muestra de los dígitos que se desean reconocer, para cumplir con esta tarea el algoritmo calcula la distancia entre el elemento a clasificar entre el resto del dataset, selecciona los K elementos más cercanos y elegir entre los k puntos la etiqueta que domine. [28]

2.1.5.13.2 Reconocimiento por Redes Neuronales

En este tipo de entrenamiento se necesita realimentar una red neuronal para que aprenda a reconocer un carácter, los caracteres son puestos en matrices de ceros y unos como se muestra en la figura 25 (a), para luego ser entrenados con datos en una red neuronal como en la figura 25(b) y pueda reconocer el carácter verdadero, un factor muy importante para el éxito o fracaso de la red neuronal en el reconocimiento de caracteres es la correcta selección de los datos para su entrenamiento. [30]

2.1.5.13.3 Reconocimiento por OCR Tesseract

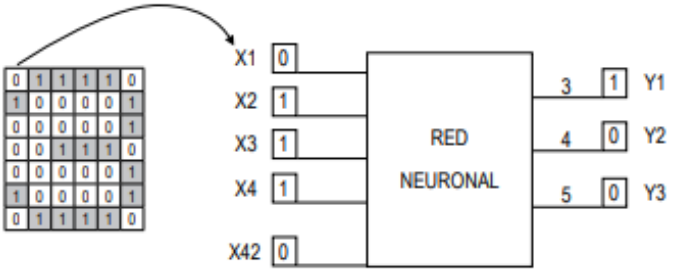
El reconocimiento por Tesseract es el más utilizado debido a su gran efectividad en el reconocimiento, utiliza la librería Pytesseract para que pueda ser utilizado en Python, se puede detectar los caracteres de distintas maneras debido que se puede cambiar un valor en la configuración del reconocimiento, algunas de las características que se obtiene es reconocer caracteres de forma vertical u obtener la mayor cantidad de texto en una cierta área detectada, el código se escribe de la siguiente manera. [28]

```
Text=pytesseract.image_to_string (placa, config= '- -psm 11')
```

El número 11 de la ecuación indica que el OCR busque la mayor cantidad de texto posible sin ningún orden en particular.

Figura 25: Detección de caracteres de una placa

a) Entrenamiento de la red neuronal

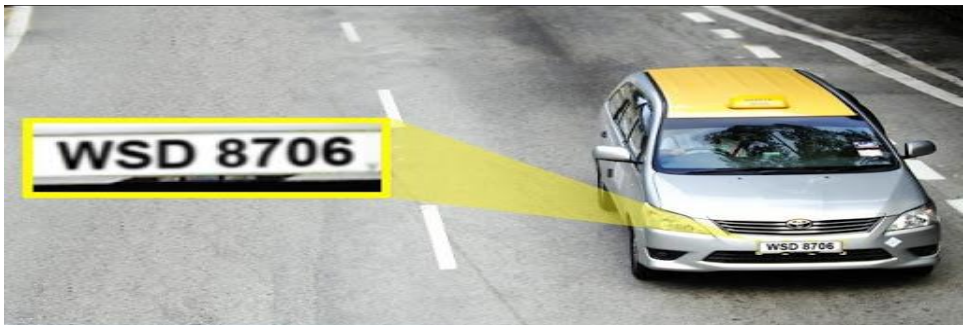


b) Datos de entrenamiento para la red neuronal

BASE DE CONOCIMIENTOS O DATOS DE ENTRENAMIENTO

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	X13	X40	X41	X42	Y1	Y2	Y3
3	1	1	1	1	1	1	0	0	0	0	0	1	0	1	1	1	1	0	0
3	0	1	1	1	1	0	1	0	0	0	0	1	0	1	1	0	1	0	0
4	1	0	0	0	0	1	1	0	0	0	0	1	1	1	0	1	0	1	0
4	0	0	0	1	1	0	0	0	1	0	1	0	0	0	1	0	0	1	0
5	1	1	1	1	1	1	1	0	0	0	0	0	1	1	1	1	0	0	1
5	1	1	1	1	1	1	1	0	0	0	0	0	1	1	1	0	0	0	1

c) Identificación de la placa del auto para reconocer y dar el dato WSD8706



Fuente: Vega H., Cortez A., Huayna A. (2009)

2.2 Marco Conceptual

En esta etapa se observará los conceptos que se utilizarán en el desarrollo del proyecto, además de las etapas de procesamiento que los autores recomiendan.

2.2.1 Definición de conceptos

2.2.1.1 Placa identificativa en fondo verde:

Es la placa diseñada por el autor del proyecto para poder simular la detección de los caracteres y recibir información de la placa en tiempo real con un video en directo.

2.2.1.2 Placa identificativa de la fábrica:

Son imágenes tomadas de la fábrica Unacem, en la cual son procesadas para obtener los caracteres de la placa.

2.2.1.3 Procesamiento de imágenes

Es el conjunto de modificaciones en una imagen digital para poder mejorar la visibilidad de características de objetos en una imagen, encargado principalmente de resaltar las características del objeto deseado. [28]

2.2.1.4 Sistema de reconocimiento:

Es un sistema encargado de reconocer un objeto de una imagen y extraer las características deseadas, en nuestro caso el sistema de reconocimiento de placas identificativas, se conforma desde la etapa de adquisición y procesamiento hasta la obtención de la información de la máquina [23]. El autor Pedro G (2013) recomienda utilizar las siguientes etapas y algoritmos para procesar y detectar a una imagen, se seguirán estas etapas en el proyecto, pero se observará si son eficientes o faltaría añadir otros procesamientos.

- **Adquisición de la imagen**

Se adquiere la imagen a través de una cámara y se guarda en una variable para poder realizar el procesamiento

- **Escala de grises**

La imagen se debe de convertir en un formato apropiado donde se pueda extraer las características. La imagen es representada con un único valor entre 0 a 255

- **Eliminación ruido**

Se utilizan filtros para reducir el ruido sal y pimiento que son puntos de color blanco o negro que dificultan el procesamiento

- **Binarización**

La imagen se convierte únicamente en dos colores, blanco y negro. Para realizar ese procesamiento se necesita tener el histograma para obtener el umbral que diferencie el rango oscuro y blanco

- **Detección de bordes**

En esta etapa se busca encontrar los bordes y formas de la imagen, se utiliza el algoritmo canny que es óptimo porque da una buena detección localización y respuesta mínima

- **Detección de contorno**

Esta etapa se encarga de encontrar todos los contornos de una imagen y guardarlo en una variable para que luego con un procesamiento se pueda discriminar el contorno deseado de las demás

- **Discriminar el contorno**

Una vez que se detectan todos los contornos, se necesita conocer cuál es el contorno deseado, por ello se utilizan distintos métodos en los cuales son la forma del contorno, el área del contorno y la longitud del contorno.

- **Mostrar los resultados**

Los resultados se deben de mostrar de manera exitosa dependiendo del objetivo del proyecto.

2.2.1.5 Código de la placa identificativa

2.2.1.5.1 Primer dígito

El primer dígito indica en que parte del proceso del cemento se encuentra la máquina, se divide en varias etapas

- Trituración (Número 2)
- Molienda de Crudo (Número 3)
- Clinkerización (Número 4)
- Molienda de Cemento (Número 5)
- Despacho (Número 6)

2.2.1.5.2 Segundo dígito

Indica el subproceso en el que se encuentra la máquina

- Recuperación y almacenamiento de Clinker (Número 1)
- Alimentación (Número 2)
- Molienda (Número 3)
- Separación (Número 4)
- Recuperación de Finos (Número 5)
- Transporte y almacenamiento de cemento (Número 6)

2.2.1.5.3 Tercer dígito

Indica las diferentes líneas que cumplen con una misma función, por ejemplo, este equivale a la sección de molienda de cemento.

- Molino 2 (Número 1)
- Molino 3 (Número 2)
- Prensa de Clinker 1 (Número 5)
- Prensa de Clinker 2 (Número 6)
- Prensa de Clinker 3 (Número 7)
- Prensa de Clinker 4 (Número 8)

2.2.1.5.4 Cuarto dígito

Indica que máquina es por ejemplos están los siguientes

- BG (Bomba de Grasa)
- PR (Prensa de Rodillos)
- HR (Horno Rotativo)
- EB (Elevador de Baldes)
- MT (Motor Eléctrico)
- RD (Reductor)
- FT (Faja transportadora)

2.2.1.5.5 Quinto dígito

Por último, el quinto digito representa en la posición que se encuentra un mismo tipo de máquina, por ejemplo

- Compuerta desviadora de fajas 555FT1 (Número 1)
- Compuerta desviadora a faja 555FT3 (Número 2)
- Compuerta desviadora a faja 551FT2 (Número 3)
- Compuerta desviadora a faja 551FT1 (Número 4)

2.3 Marco Metodológico

En esta sección se realiza una explicación del procedimiento para realizar la detección de la placa identificativa.

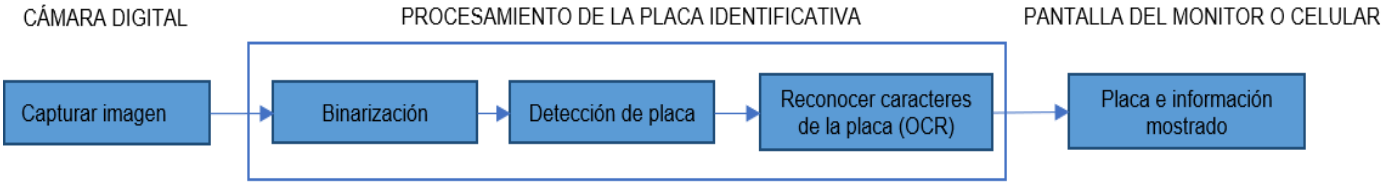
2.3.1 Métodos

Para el desarrollo del procesamiento de la imagen, se ha considerado realizar el análisis, según aplique a la etapa, siguiendo los siguientes métodos.

- Modelamiento por diagrama de bloques
- Modelamiento por diagrama pictórico
- Modelamiento por diagrama de flujo
- Procesamiento de la placa identificativa

2.3.1.1 Diagrama de bloques:

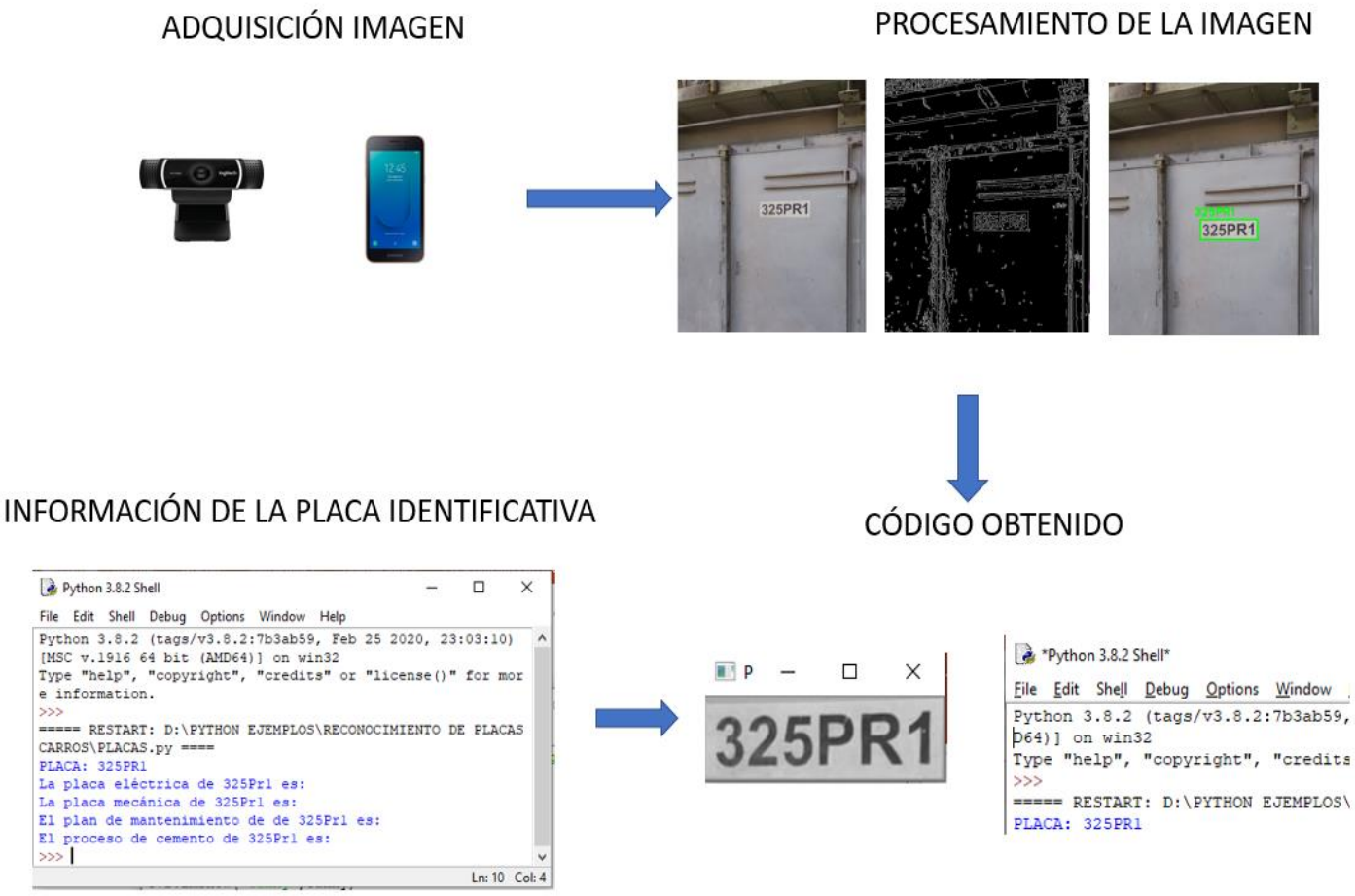
Figura 26: Diagrama de bloques



Fuente: Elaboración Propia

2.3.1.2 Diagrama pictórico

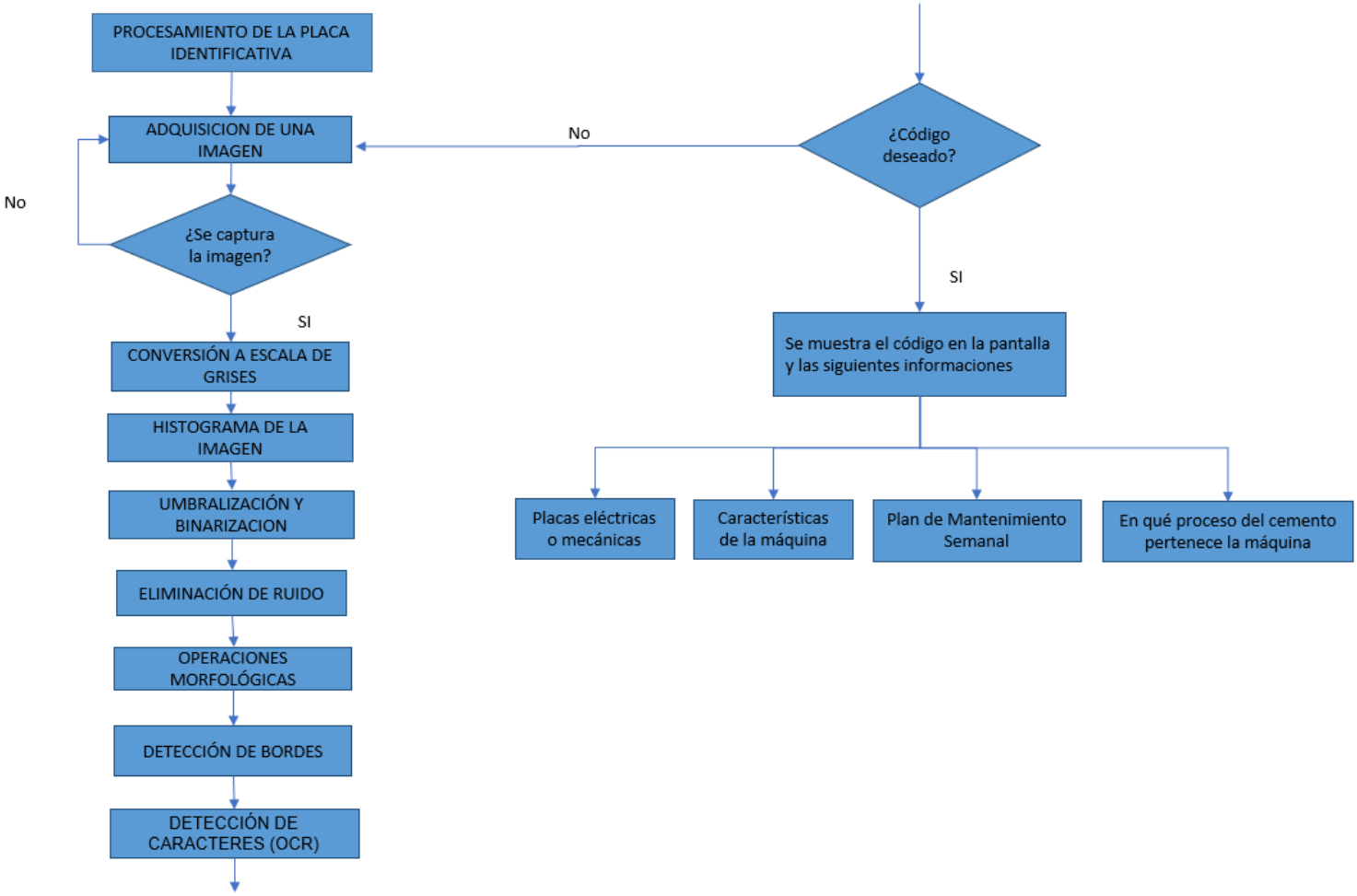
Figura 27: Diagrama pictórico



Fuente: Elaboración Propia

2.3.1.3 Diagrama de flujos

Figura 28: Diagrama de flujos



Fuente: Elaboración Propia

2.3.1.4 Procedimiento para detectar la placa identificativa

Para poder obtener la información deseada de una máquina se realiza el procesamiento de la placa identificativa de la siguiente manera.

- **Adquisición de la imagen**

  Para poder adquirir una imagen se necesita capturar la imagen a través de una cámara, por lo cual se identifica la cámara y se realiza la captura en tiempo real de la imagen. En este paso se utiliza el reconocimiento en tiempo real para las placas identificativas en fondo verde y reconocimiento de imágenes unitarias para las placas de la fábrica.

- **Iluminación de la imagen**

  Para poder obtener una buena imagen se necesita el histograma RGB de la imagen que nos muestra las intensidades en la imagen.

- **Interfaz de la aplicación**

  Se muestra el interfaz que se diseña con botones (Video, mostrar, capturar) para poder observar la imagen procesada además de como se muestra el video en la interfaz.

- **Convertir la imagen a grises**

  Una vez que se obtiene la imagen en tiempo real se toma un fotograma y se realiza operaciones para poder discriminar la placa identificativa de la imagen completa por ello el primer paso es convertirlo a escala de grises.

- **Eliminar el ruido de la imagen**

  Se encarga de eliminar el mayor ruido posible para evitar que la placa no se encierre o existan falsos contornos.

- **Conversión a binario**

  Una vez obtenido la imagen en escala de grises y obtenido el histograma se realiza la umbralización que consiste en ver cuál es la intensidad de valor mínimo que separa el rango de oscuros y claros para poder convertirle en una imagen binaria.

- **Detección de bordes Canny**

  En esta etapa se encarga de detectar los bordes que se encuentran en toda la imagen para que luego se pueda procesar cada una.

- **Operaciones morfológicas**

  Una vez que se obtiene la imagen en binaria se pasa a realizar operaciones morfológicas como dilatar o erosionar la imagen con el objetivo de que la placa esta encerrada para que pueda ser procesada.

- **Encontrar y dibujar los contornos**

  Se realiza la detección de los contornos de toda la imagen para poder procesar la imagen y ver los contornos encerrados en el cual uno de esos contornos pertenece a la placa identificativa.

- **Reconocer el contorno deseado**

  Una vez que se detectó todos los contornos, se pasa a seleccionar el contorno deseado que cumpla con las características obtenidas y así poder mostrar el contorno en tiempo real, por ello se reconoce el área que encierra en cada contorno y que tenga forma rectangular, realizando una aproximación del área detectado según la experiencia de detección.

- **Reconocer y mostrar los caracteres de la placa**

Una vez que se dibuja el contorno se puede observar la placa identificativa y se realiza la detección de caracteres para poder descifrar las letras que encierra la placa, además se muestra con un cartel encima de la placa identificada.

- **Mostrar la última placa capturada**

  En esta etapa se muestra la última placa que se ha capturado cuando se presiona el botón mostrar información.

- **Eliminar los caracteres especiales**

  Es esencial esta etapa que elimina los caracteres especiales como el signo de interrogación. Que debido al ruido o desenfoque provoca que el reconocimiento de caracteres detecte códigos con más caracteres.

- **Mostrar información de la base de datos**

  Se muestra toda la información que se desea obtener de una máquina que se encuentra en la base de datos.

2.3.2 Sistema de reconocimiento

2.3.2.1 Ancho y altura de la placa

La altura de la placa es 175mm de ancho y el de altura es 65mm

2.3.2.2 Captura de la placa identificativa de color verde

Se realiza la captura de la placa a 1.5 metros de distancia, de manera frontal capturando únicamente la placa con el fondo uniforme, tener un histograma con una iluminación con buen contraste para poder realizar el procesamiento.

2.3.2.3 Captura de la placa identificativa de la fábrica

Se realiza la captura de la imagen a unos 2 metros de distancia, de manera frontal, capturando la placa identificativa junto a otros elementos, las imágenes deben de tener un histograma que favorezca el procesamiento.

2.3.2.4 Eficiencia

Para saber la eficiencia del sistema de reconocimiento de placas, se realizará una captura de 100 imágenes y se observará el resultado de los códigos reconocidos, si se obtiene un porcentaje exitoso mayor a 90% indicaría que el sistema es eficiente, además la información de la máquina que da como resultado debe de ser la correcta, para obtener la eficiencia se utilizará el método prueba de hipótesis para una proporción con cola derecha, el cual se presentan las siguientes variables.

P: Probabilidad de éxito del reconocimiento de la placa del proyecto

Po: Probabilidad mínima que el proyecto debe de tener (90%)

n: Numero de muestras

$\sigma_{\bar{P}}$: Desviación estándar de la distribución

Z: Número de desviaciones estándar

$\bar{P}$: Proporción muestral

Para obtener la eficiencia se realiza los siguientes pasos.

Paso1: Proponer la hipótesis

H_0: P_0=0.09

H_1: P >0.09; $\boldsymbol{\bar{P}} = \frac{\boldsymbol{Exito}}{\boldsymbol{Muestras}}$

Paso 2: Especificación de la significación:

El nivel de confianza es 90% por lo tanto ($1 - \alpha = 90\%$) y la significación es ($\alpha = 10\%$)

Por tabla obtenemos la Z cola derecha de la probabilidad.

Tabla1: Valores de Z con distintos niveles de confianza

Confianza $(1-\alpha)$	Significación (α)	Izquierda	Derecha
90%	10%	Z= -1.28	Z= 1.28
95%	5%	Z= -1.64	Z= 1.64
99%	1%	Z= -2.33	Z= 2.33

Fuente: Elaboración propia

Paso3: Halla los valores críticos y de prueba

Vc: Zc=1.28

Vp: $Z = \frac{\bar{P}-P_0}{\sigma_{\bar{P}}}$; $\sigma_{\bar{P}} = \sqrt{\frac{P_0*(1-P_0)}{n}}$

Paso4: Establecer zona de aceptación/Rechazo de Ho

Paso5: Decisión y conclusión de la prueba.

- Decisión: Se acepta o rechaza H0
- Conclusión: Se puede o no afirmar que la proporción es mayor a la probabilidad de éxito 90%.

2.3.2.5 Plan de gestión de los riesgos

En la tabla2 se muestra los riesgos que evitan el cumplimiento del proyecto en el plazo establecido.

Tabla2: Áreas encargadas de los Riesgos del proyecto

RIESGOS DEL PROYECTO	
Categoría	**Factor de Riesgo**
Administración	• **Alcance y entregable del proyecto** • **Ampliación del Cronograma** • **Cambios en el Alcance** • **Roles y Responsabilidades no definidas íntegramente** • **Calidad inadecuada en el Plan del Proyecto**
Recursos	• **Deficiencia en la asignación de Recursos** • **Habilidades del equipo del proyecto** • **No disponibilidad en algún bien o servicio**
Organizacionales	• **Falta de priorización del Proyecto** • **Fondos inadecuados o interrumpidos** • **Deficiencia en la definición del alcance**
Comunicación	• **Falta de dirección en el Proyecto** • **Falta de comunicación entre los miembros del Proyecto** • **Comunicación ambigua o escasa**
Externos	• **Cambio de prioridad del dueño** • **Riesgos del País, terremoto, clima, pandemia**

Fuente: Elaboración propia

2.3.2.5.1 ROLES Y RESPONSABILIDADES

Tabla3: Roles de los Riesgos del proyecto

Roles para el manejo de Riesgos	Funciones
Originador del Riesgo	• La temprana identificación del riesgo dentro del Proyecto • Documentación formal del riesgo, completando el formato de riesgos • La publicación del formato de Riesgo para la revisión del Gerente del Proyecto
Gerente del Proyecto	• Recibir todas las notificaciones de los riesgos • Analizar los riesgos para ingresarlos al Registro • Reportar y comunicar las decisiones tomadas • Monitorear los riesgos ya Ingresados.
Grupo Revisión del Proyecto	• Identificación de solicitud de cambio para mitigar los riesgos • Asignación de acciones para mitigar el riesgo • El cierre de riesgo que no presentan acciones pendientes y no presentan más impacto al proyecto
Equipo del Proyecto	• Mitigar el Riesgo delegados por el grupo de revisión del Proyecto

Fuente: Elaboración propia

2.3.2.5.2 Probabilidad e Impacto de los Riesgos

Se muestra que tan probable es que se genere el riesgo, además del impacto que genera el riesgo en el proyecto, en la tabla 4 se observar la probabilidad de los riesgos

Tabla4: Probabilidad de los riesgos

PROBABILIDAD			
TIPOS DE PROBABILIDAD	DEFINICIÓN	DETALLE	VALOR
Muy Probable	Se espera que ocurra en la mayor de las circunstancias	Probabilidad de ocurrencia mayor a 75%	5

Probable	El evento ocurre a menudo	Probabilidad de ocurrencia del 50% a 75%	4
Posible	El evento ocurre a veces	Probabilidad de ocurrencia del 25% al 49%	3
Poco Probable	El evento es posible pero ocurre raramante	Probabilidad menor de 25%	2
Improbable	La ocurrencia del evento es posible pero nunca a ocurrido o muy escasa vez	No es posible que ocurra en los proximos años	1

Fuente: Elaboración propia

En la Tabla5 se observa el nivel de impacto que generan los riesgos.

Tabla5: Impacto de los riesgos

IMPACTO		
CATEGORÍA	**VALOR**	**DESCRIPCIÓN**
Catastrófico	**5**	La mayor parte de la operación se interrumpe
Muy Grave	**4**	Perdida temporal de la funcionalidad
Grave	**3**	Interrupcion moderada a las actividades
Menor	**2**	Poca interrupción a las actividades
Leve	**1**	No afecta las actividades dia a dia

Fuente: Elaboración propia

2.3.2.5.3 Nivel de los Riesgos

Para establecer cuáles son los riesgos que afectan al proyecto se realiza una categorización que muestre el nivel de riesgo.

La fórmula para ver el nivel de Riesgo es:

Nivel de Riesgo= (Probabilidad x Impacto)

Se obtiene la siguiente matriz de Riesgo

Riesgo alto **Riesgo Moderado** **Riesgo Bajo**

Probabilidad						
5	**5**	**10**	**15**	**20**	**25**	
4	**4**	**8**	**12**	**16**	**20**	
3	**3**	**6**	**9**	**12**	**15**	
2	**2**	**4**	**6**	**8**	**10**	
1	**1**	**2**	**3**	**4**	**5**	
0	**1**	**2**	**3**	**4**	**5**	**IMPACTO**

2.3.2.5.4 CATEGORÍAS DE RIESGOS

2.3.2.5.4.1 RIESGOS EN EL CUMPLIMIENTO DEL PROYECTO

Son riesgos que evitan que el proyecto se cumpla en el plazo establecido

Tabla6: Riesgos que evitan el cumplimiento del proyecto

Código	Factor de Riesgo	Descripción de Impacto	Probabilidad	Impacto	Nivel	Posible Respuesta	Acción Preventiva	Responsable
R01	Alcance y entregable del proyecto	Impacto en la entrega estimada del proyecto	3	5	15	Evitar	Realizar la revisión del Alcance y entregable estableciendo el plazo de entrega	Equipo del Proyecto
R02	Ampliación del Cronograma	Impacto en la planificación y costos en implementar el proyecto	3	3	9	Evitar	Implementar el proyecto según lo planificado	Gerente del Proyecto
R03	Cambios en el Alcance	Impacto en costos y cronograma	2	3	6	Transferir	Registrar los cambios del alcance para mejorar el proyecto con una nueva planificación	Equipo del Proyecto
R04	Roles y Responsabilidades no definidas íntegramente	Impacto en la organización por falta de seriedad	1	4	4	Transferir	Revisión de Rol y responsabilidad definida para el proyecto, modificar o cambiar responsables	Gerente del Proyecto

Código	Factor de Riesgo	Descripción de Impacto	Probabilidad	Impacto	Nivel	Posible Respuesta	Acción Preventiva	Responsable
R05	Calidad inadecuada en el Plan del Proyecto	Impacto en la implementación final del proyecto	4	4	16	Evitar	Modificar los entregables en base a los estándares definidos y exigir el cumplimiento al equipo de trabajo	Gerente del Proyecto
R06	Deficiencia en la asignación de Recursos	Impacto en la planificación de tiempo	1	2	2	Retener	Readecuar el equipo de trabajo en el proyecto, redistribuir las tareas	Gerente del Proyecto
R07	Habilidades del equipo del proyecto	Impacto en el tiempo de ejecución del proyecto	1	2	2	Retener	Selección de personal para el desarrollo del proyecto	Gerente del Proyecto
R08	No disponibilidad en algún bien o servicio	Incongruencia en lo planificado e implementado	3	4	12	Evitar	Planificar las actividades que se necesitan los bienes y servicios	Gerente del Proyecto
R09	Deficiencia en la definición del alcance	No se cumple los objetivos y las necesidades del usuario	1	4	4	Transferir	El alcance debe ser registrado en un lenguaje claro y sin ambigüedades	Gerente del Proyecto
R10	Falta de priorización del Proyecto	Retrasos en el cronograma y problema de costos	2	3	6	Transferir	Consensuar con el usuario el registro de la planificación anual de la empresa y los tiempos para implementar el proyecto	Gerente del Proyecto

Código	Factor de Riesgo	Descripción de Impacto	Probabilidad	Impacto	Nivel	Posible Respuesta	Acción Preventiva	Responsable
R11	Fondos inadecuados o interrumpidos	Adicion en el tiempo de ejecución	3	5	15	Evitar	Establece restricciones de presupuesto necesario para el cumplimiento	Gerente del Proyecto
R12	Falta de dirección en el Proyecto	Retraso y problemas con los resultados	3	5	15	Evitar	Establecer los estándares requeridos, los canales de comunicación, roles, responsabilidades al equipo del proyecto	Gerente del Proyecto
R13	Falta de comunicación entre los miembros del Proyecto	Incompatibilidad en ideas e implica en rehacer tareas	2	5	10	Transferir	Descentralizar la asignación de tareas y dar a conocer la documentación para que el equipo trabaje autónomamente	Gerente del Proyecto
R14	Cambio de prioridad del dueño	Cambio giro o redefinición de mercado o operaciones	1	5	5	Transferir	Establecer este tópico en las restricciones del proyecto	Equipo del Proyecto
R15	Riesgos del Pais, terremoto, clima, pandemia	Impacto en los tiempos de implementación por falta de conectividad en el equipo	1	5	5	Retener	Proveer un medio de comunicación adicional si existen situaciones fortuitas	Equipo del Proyecto

Fuente: Elaboración propia

2.3.2.5.4.2 RIESGOS EN EL FUNCIONAMIENTO DEL PROYECTO

Son Riesgos que provocan que el proyecto no funcione correctamente

Tabla7: Riesgos que provocan el mal funcionamiento del proyecto

ID RIESGO	Factor de Riesgo	Descripción del impacto	Probabilidad	Impacto	Nivel	Acción Preventiva	Responsable
R16	Suciedad en la placa de la máquina	Afecta el procesamiento de la imagen	4	4	16	Antes de realizar la captura observar si la placa del código se encuentra limpia	Originador del Riesgo
R17	Mala captura de la imagen a una distancia no recomendada	Mal procesamiento de la imagen	4	4	16	Enseñar a los empleados la buena utilización del proyecto	Originador del Riesgo
R18	Desactualización de información en la máquina	Información errónea de la máquina	2	5	10	Responsable en actualizar la información pendiente que todas las máquinas tengan información actualizada	Equipo del proyecto
R19	Fallos en el servidor Web	No se recibe información de las máquinas	2	5	10	Responsable de los servidores pendiente en el funcionamiento del servidor Web	Equipo del proyecto

Fuente: Elaboración propia

CAPÍTULO 3

DESARROLLO DE LA SOLUCIÓN

En este capítulo se presenta las herramientas que se utilizan para desarrollar el proyecto y la solución del sistema de reconocimiento de placas identificativas de las máquinas correspondientes respecto al fondo de la imagen. La solución del sistema de reconocimiento muestra información desde la etapa de adquisición de imagen hasta la obtención de información de la placa.

Se realizan pruebas con un programa realizado en Python, con imágenes procesadas en tiempo real desde placas identificativas con un fondo de color verde e imágenes capturados desde la misma fábrica UNACEM para ver la diferencia entre los dos procesamientos.

Además, se muestra el código fuente de las rutinas realizadas en PYTHON siguiendo los algoritmos que se presentan en capítulos anteriores.

3.1 Prioridades de cumplimiento

Nos indica que tan eficiente es el cumplimiento de la tecnología respecto a los requerimientos del proyecto.

ALTA: Cumple a gran escala las expectativas del proyecto y a un bajo costo

MEDIA: Cumple, con las expectativas del proyecto y el costo es aceptable

BAJA: Cumple, pero con probabilidad alto de dar errores y a un costo alto

NO CUMPLE: No cumple con el requerimiento del proyecto

3.2 Selección de tecnología

3.2.1 Digitalización de código

3.2.1.1 Requerimientos del proyecto

Los criterios que se buscan para que el proyecto funcione bien son:

- **Reconocimiento:** Detección correcta de los caracteres, mínimo 90% eficiencia
- **Rapidez:** Detección rápida de los caracteres no mayor a 2 segundos
- **Distancia mínima:** Detección a una distancia mínima de 0.5 metro debido a la precaución o a la accesibilidad de la máquina
- **Instalación:** Tecnología con instalación accesible en cada máquina de la fábrica

3.2.1.2 Alternativas tecnológicas

Existen distintos tipos de tecnología para obtener el código digitalizado de los caracteres de una placa, como la tecnología NFC, Codigo QR o Visión artificial, en las siguientes tablas8, 9, 10 se mostrarán si las tecnologías cumplen con los requerimientos del proyecto.

Tabla8: Tecnología NFC

	Tecnología NFC	**Prioridad**
Reconocimiento	Buen reconocimiento de los caracteres con eficiencia mayores al 90%	ALTA
Rapidez	Detección rápida de los caracteres, con milisegundos de respuesta	ALTA
Distancia Mínima	Detección cercana, no mayor a 10 o 20 cm	NO CUMPLE
Instalación	Se necesita instalar en cada placa de la fábrica un código NFC para poder reconocer el código, generando costo de personal y tiempo de instalación	BAJA

Fuente: Elaboración propia

Tabla9: Tecnología por código QR

	Código QR	**Prioridad**
Reconocimiento	Buen reconocimiento de los caracteres según el enfoque del código QR con eficiencia mayores al 90%	ALTA
Rapidez	Detección rápida de los caracteres, con milisegundos de respuesta	ALTA
Distancia Mínima	Detección cercana mayor a 0.5m	MEDIA
Instalación	Se necesita poner en cada máquina un código QR para poder reconocer el código, generando costo y tiempo de instalación	BAJA

Fuente: Elaboración propia

Tabla10: Tecnología por Visión artificial

	Visión Artificial	Prioridad
Reconocimiento	Buen reconocimiento de los caracteres según el procesamiento con eficiencia mayores al 90%	ALTA
Rapidez	Detección rápida de los caracteres, con milisegundos de respuesta	ALTA
Distancia Mínima	Detecta el código a distancias mayores de 0.5m, dependiendo del procesamiento de la imagen	ALTA
Instalación	No hay necesidad de instalar debido que todas las máquinas de la fábrica tienen su propia placa con código identificativo que puede ser procesado con esta tecnología	ALTA

Fuente: Elaboración propia

Se elige la tecnología por Visión artificial, debido que cumple con todos los requerimientos del proyecto como se muestra en la tabla.

3.3 Herramientas del proyecto

3.3.1 Herramientas para procesar placa con fondo uniforme

3.3.1.1 Cámara digital (WEB)

3.3.1.1.1 Consideraciones Técnicas

Se necesita que la cámara cumpla con los requerimientos mínimos para poder realizar el proyecto.

- **Pixeles**: 1920x1080 para una buena captura de la imagen
- **Resolución de pantalla:** Resolución mínima de 12MP
- **Enfoque:** Se necesita que la cámara enfoque de manera automática
- **Sensor:** Alta Definición Completa, Tecnología CMOS O CCS
- **Distancia Focal:** No será necesario utilzar el Zoom debido que la distancia estará predeterminada por 1.5m de Distancia

- **Apertura Focal:** Apertura focal mínima de F/2.5 para que el sensor reciba la luz necesaria para capturar una buena imagen, mientras menor sea el número que acompaña a F, mejor será la recepción de luz

3.3.1.1.2 Alternativas

Se compara a través de la tabla las siguientes marcas de cámaras web y se elige la que cumple con los requerimientos mínimos del proyecto

Tabla11: Especificaciones de cámaras Web

CAMARA WEB	**Logitech C922 Pro**	**Micronics Othelo W360**	**Microsoft Lifecam HD 3000**	**Genius WIDECAM F100**
Pixeles	1920 x 1080	1280 x 800	1280 x 800	1920 x 1080
Sensor	CMOS	CMOS	CMOS	CMOS
Lente de Cristal	Si	No	Si	No
Resolucion de Captura	12MP	3MP	5MP	12MP
Video	1080p/30fps	720p/30fps	720p/30fps	1080p/30fps
Tipo de Enfoque	Automático	Automático	Automático	Manual
Tipo	Color	Color	Color	Color
Apertura Focal	F/2.0	F/2.5	F/2.5	F/2.0

Fuente: Elaboración propia

Se utiliza la cámara Web (streaming C922 Hd 720p a 60 Fps) debido que cumple con los requerimientos mínimos, además que brinda alta calidad en el reconocimiento en tiempo real del procesamiento de la placa identificativa en fondo verde, en la siguiente tabla se observa el cumplimiento de los requerimientos mínimos.

Tabla12: Especificaciones de la Cámara Web C922 Pro

Especificaciones	**Valor**	**Prioridad**
Pixeles	1920 x 1080	Alta
Resolución de pantalla	12MP	Alta
Tipo de enfoque	Automático	Alta
Sensor	Full HD, Tecnología CMOS	Alta
Distancia Focal	No es necesario debido que esta predefinido por 1.5m	Media
Apertura Focal	F/2.0	Alta

Fuente: Elaboración Propia

En la siguiente figura se muestra la imagen de la cámara Web Logitech C922 Pro

Figura 29: Logitech Cámara Web

Fuente: Elaboración propia

3.3.1.2 Computadora

3.3.1.2.1 Consideraciones Técnicas

Se necesita que la computadora cumpla con los requerimientos mínimos para poder realizar el proyecto.

- **Procesador:** Mínimo 2 núcleos para un procesamiento estable
- **Memoria Ram:** Memoría Ram mínima de 8Gb
- **Sistema:** Windows 10 de 64 bits para compatibilidad segura con el programa
- **GPU:** Buena resolución y velocidad de ejecución mínimo de 1kHZ de frecuencia

3.3.1.2.2 Elección del equipo

La computadora en donde se va a procesar la imagen de la placa identificativa tiene las siguientes características que cumple con los requisitos mínimos.

Figura 30: Computadora donde se procesa las imágenes

Fuente: Elaboración propia

Tabla 13: Especificaciones de la computadora

Especificaciones	Valor	Prioridad
Procesador	AMD Ryzen 5 2600 Six Core	Alta
Memoria Ram	16Gb	Alta
Sistema	Windows 10 de 64 Bits	Alta
Tarjeta Video	Geforge1060GTX 3Gb	Alta

Fuente: Elaboración propia

3.3.2 Herramienta para procesar placa de la fábrica (Fondo no uniforme)

3.3.2.1 Celular

3.3.2.1.1 Consideraciones técnicas

Se necesita que el equipo cumpla con los requerimientos de la cámara y del Hardware

3.3.2.1.1.1 Consideraciones Técnicas de la cámara frontal

- **Resolución de pantalla**: Resolución mínima de 12MP
- **Enfoque:** Se necesita que la cámara enfoque de manera automática
- **Sensor:** Alta Definición Completa FULL HD, Tecnología CMOS O CCS
- **Distancia Focal:** No será necesario utilizar el Zoom debido que la distancia estará predeterminada por 1.5m de Distancia
- **Apertura Focal:** Apertura focal mínima de F/2.5 para que el sensor reciba la luz necesaria para capturar una buena imagen, mientras menor sea el número que acompaña a F, mejor será la recepción de luz

3.3.2.1.1.2 Consideraciones técnicas del Hardware

- **Procesador:** Mínimo 2 nucleos para un procesamiento estable
- **Memoria Ram:** Memoría Ram mínima de 1Gb
- **Sistema:** Sistema operativo Android V5.0, 5.1, 5.1.1 compatibilidad segura con el programa
- **GPU:** Buena resolución y velocidad de ejecución mínimo de 1kHZ de frecuencia

3.3.2.1.1.3 Alternativas

Se compara a través de la tabla 14 las siguientes marcas de celulares según su cámara y Hardware y se elige la que cumple con los requerimientos mínimos del proyecto

Tabla14: Especificaciones de celulares

Celulares		**Samsung Galaxy J5**	**LG G5 LTE H844**	**Motorola Moto G4 Plus**	**Alcatel Idol4**
Cámara	Resolución de Captura	12MP	16MP	16MP	13MP
	Tipo de Enfoque	Automático	Automático	Automático	Automático
	Sensor	CMOS	CMOS	CMOS	CMOS
	Apertura Focal	F/1.9	F/1.8	F/2.0	F/2.0
	Tipo	Color	Color	Color	Color
	Resolución pantalla	720x1280 /294ppi	1440x2560 / 554ppi	1080x1920 /401ppi	1080x1920 /424ppi
	Distancia Focal	3.7mm	4.4 mm	4.68mm	No especifica
Hardware	Procesador	ARM Cortex-A53 (4 Nucleos)	Qualcomm Snapdragon 820 (4 Nucleos)	Qualcomm Snapdragon 617 (8 Nucleos)	Qualcomm Snapdragon 617 (8 Nucleos)
	Memoria Ram	1.5Gb	4 Gb	4Gb	2Gb
	Sistema	Android 5.1.1 Lollipo	Android 6.0.1 Marshmallow	Android 6.0.1 Marshmallow	Android 6.0.1 Marshmallow
	Tarjeta Video	Qualcomm Adreno 306 (400Mhz de Frecuencia)	Qualcomm Adreno 530 (624Mhz de Frecuencia)	Qualcomm Adreno 405 (550Mhz de Frecuencia)	Qualcomm Adreno 405 (550Mhz de Frecuencia)

Fuente: Elaboración propia

3.3.2.1.1.4 Selección

Se observa que todos cumplen con los requerimientos básicos por lo +

cual se escoge el que tiene menor costo que es el Samsung Galaxy J5, en la siguiente tabla 15 se observa el cumplimiento de los requisitos mínimos.

Tabla 15: Especificaciones del celular Galaxy J5

Especificaciones		**Valor**	**Prioridad**
Cámara	Resolución de pantalla	12MP	Alta
	Tipo de enfoque	Automático	Alta
	Sensor	Full HD, Tecnología CMOS	Alta
	Distancia Focal	3.7mm	Alta
	Apertura Focal	F/1.9	Alta
Hardware	Procesador	ARM Cortex-A53 (4 Nucleos)	Alta
	Memoria Ram	1.5Gb	Media
	Sistema	Android 5.1.1 Lollipo	Media
	Tarjeta Video	Qualcomm Adreno 306 (400Mhz de Frecuencia)	Alta

Fuente: Elaboración Propia

En la siguiente figura 30 se muestra el equipo seleccionado para el proyecto

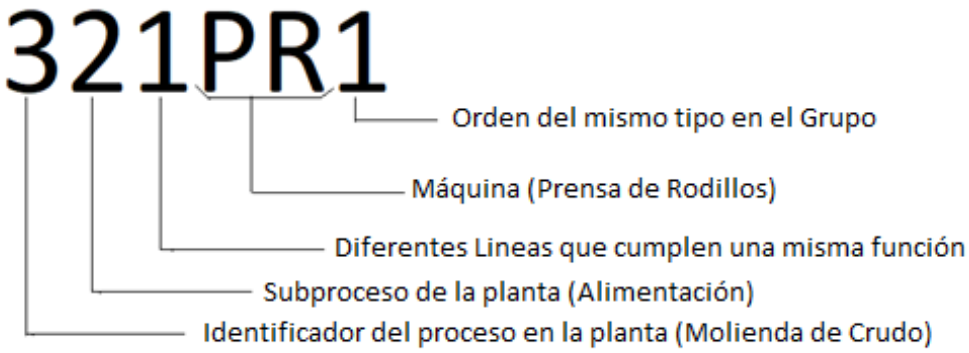
Figura 31: Celular Galaxy J5 (16G) SM-J500M

Fuente: Elaboración propia

3.4 Código de la placa de una máquina

La placa identificativa de una máquina tiene un código que los identifica, en el cual cada una tiene su significado como se observa en la figura 32

Figura 32: Placa identificativa de la fábrica



Fuente: Elaboración Propia

3.5 Programa

Para elegir el programa necesitamos observar la siguiente tabla que muestran las ventajas de cada uno y así poder elegir el mejor para el proyecto. Debe de cumplir con los siguientes requisitos

- El lenguaje tiene que ser de código abierto
- Debe de tener una librería amplia en visión artificial
- El programa debe de ser de bajo costo
- Debe de poder crear ejecutables para el uso en celulares

Tabla16: Programas para desarrollar el proyecto

Programa	PYTHON	MATLAB
Lenguaje de programación	Python,C/C++	LenguajeM
Código abierto	Si	No
Librería	Amplio	Limitado
Costo del programa	Gratis	Elevado
Compatibilidad	Windows Linux Android y Mac OS	Windows Linux Android y Mac OS
Portable	Si	No

Fuente: Elaboración Propia

Se elige el programa Python por lo siguiente

- Es de código abierto lo que permite realizar modificaciones y facilidad de intercambiar ideas
- El programa es gratuito
- Es portable y se puede convertir el formato para uso en celulares

3.6 Lenguaje de programación (Python)

Se observa una tabla que diferenciará entre la elección del lenguaje de programación Python y el lenguaje C++, tiene que cumplir con los siguientes requisitos

- Amplia variedad en bibliotecas para el uso de visión artificial
- Facilidad de lectura y escritura
- Debe de ser de sintaxis simple
- El lenguaje debe de ser dinámico

Tabla17: Lenguajes de programación

Lenguaje de programación	Python	C++
Librerías	Amplia en inteligencia artificial, compatible con OpenCV	Moderado en inteligencia artificial, compatible con OpenCV
Comunidad	Lenguaje dinámico y alta comunidad	Lenguaje dinámico y alta comunidad
Sintaxis	Simple	Complejo
Escritura y Lectura	Facilidad	Facilidad
Aprendizaje	Sencillo	Moderado

Fuente: Elaboración Propia

Se eligió el lenguaje de programación Python por las siguientes razones:

- Contiene una gran cantidad de bibliotecas como Numpy, Scipy, Pybrain, OpenCv, Pytesserac y Tkinter que se utilizan en el aprendizaje automático, inteligencia artificial y procesamiento de imágenes.
- Tiene una comunidad mayor y es un lenguaje dinámico.
- Es de sintaxis simple y de facilidad de lectura lo que le hace un lenguaje adecuado para el procesamiento de la imagen.

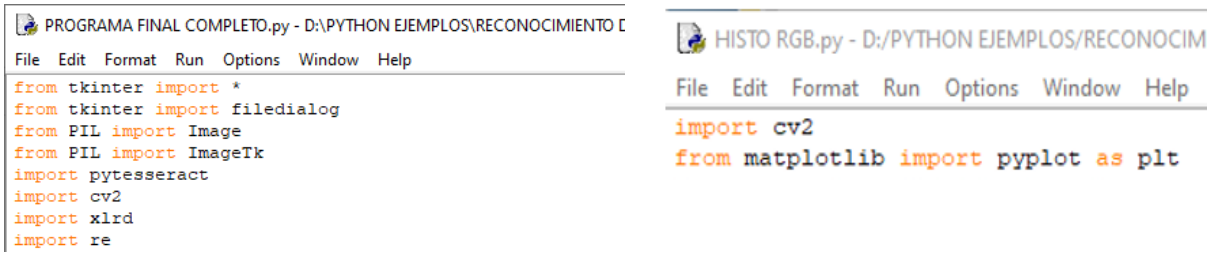
3.7 Librerías de la aplicación

Se utilizarán las siguientes librerías para el desarrollo de la aplicación.

- Pytesseract: Se elige la librería pytesseract (Versión 0.3.3) debido que es la encargada para que Python pueda utilizar el reconocimiento de caracteres OCR con Tesseract (Versión 5.0.0). Se utiliza después de reconocer el contorno de la placa identificativa para reconocer los caracteres de la placa.
- OpenCv: Se elige la librería OpenCv (Versión 4.3.0) principal para poder realizar el procesamiento de la imagen debido que contiene códigos utilizados para la visión artificial de manera gratuita. Se utiliza la versión OpenCv.
- Tkinter: Se elige la librería Tkinter (Versión 8.6) debido que es una librería encargada de poder realizar interfaces que se puede interactuar entre el usuario y el programa, se utilizará para poder capturar la imagen con un botón y poder simular la interfaz que muestra la información de la placa.
- PIL: Es utilizado en la (Versión 7.2.0) para importar Image y ImageTk donde nos permitirá mostrar las imágenes a través de la interfaz gráfica Tkinter
- Xlrd: Es utilizado la librería Xlrd (Versión 1.2.0) para poder llamar la base de datos creados en el programa Excel y poder mostrar la información requerida de la máquina detectada
- Matplotlib: Es utilizado en la (Versión 3.4.0) para importar pyplot que nos permite graficar los histogramas de una imagen ya se en grises o en BGR
- Re: Es utilizada en la (Versión 2.2.1) únicamente para filtrar los caracteres en minúsculas

Para poder importar las librerías se presenta el siguiente código que se muestra en la figura 33. Donde se importa para el sistema de reconocimiento y para la obtención del histograma respectivamente.

Figura 33: Importación de Librerías



Fuente: Elaboración Propia

3.8 Procedimiento para detectar la placa identificativa

3.8.1 Adquisición de la imagen

Para la adquisición de la imagen en tiempo real se utiliza un código que muestra el video en directo guardado en una variable llamado Cap.

Cap=cv2.VideoCapture(0, cv2.CAP_DSHOW)

Esta variable Cap contiene 2 subvariables llamados ret, frame, cuando se inicia la captura de la imagen la variable ret se vuelve TRUE y si aún no se ha iniciado se vuelve FALSE, la variable frame equivale a los fotogramas tomados del video. En la siguiente figura 34 se muestra el subproceso creado llamado video_de_entrada () en el cual es llamado por una interfaz gráfica que se explicará más adelante, la función esta encargada en mostrar las imágenes del video y activar los botones de las interfaces, además se muestra una parte del subproceso visualizar () donde se guardan los fotogramas en la variable frame para poder ser procesados.

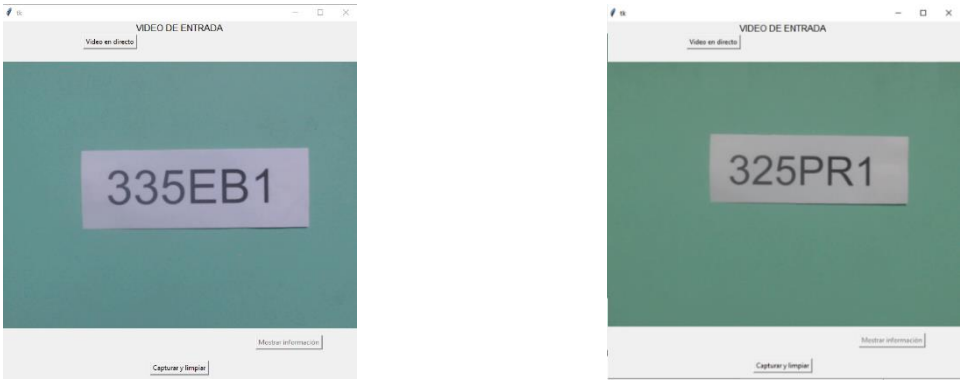
Figura 34: Función para observar el video en directo

```
def video_de_entrada():
    global cap

    Boton3.configure(state="active")
    Boton2.configure(state="disabled")
    Boton1.configure(state="active")
    InfVideo.configure(text="")
    cap = cv2.VideoCapture(0, cv2.CAP_DSHOW)
    visualizar()

def visualizar():
    global cap
    global im
    ret, frame = cap.read()
```

Fuente: Elaboración Propia

En la siguiente figura 35 se muestra las imágenes adquiridas del video en directo de las placas identificativas en fondo verde.

Figura 35: Placas identificativas en fondo verde mostradas en directo



Fuente: Elaboración Propia

Para el procesamiento de las placas identificativas de la fábrica Unacem se utiliza el código cv2.imread(name.jpg) donde name es la imagen en formato jpg que se necesita leer.

En la siguiente figura 36 se muestran las imágenes capturadas de la fábrica UNACEM.

Figura 36: Placas identificativas de la fábrica mostrados como imagen

Fuente: Elaboración Propia

3.8.2 Iluminación de la imagen

Una imagen debe de tener una buena iluminación para poder realizar un buen procesamiento, se tiene que demostrar a través del histograma que las intensidades se encuentran en la parte central, para obtener el histograma se realiza el siguiente código mostrado en la figura 37.

Figura 37: Código para obtener el histograma de la imagen

HISTO RGB.py - D:/PYTHON EJEMPLOS/RECONOCIMIENTO DE PLACAS/HISTOGRAMAS Y CA...

File Edit Format Run Options Window Help

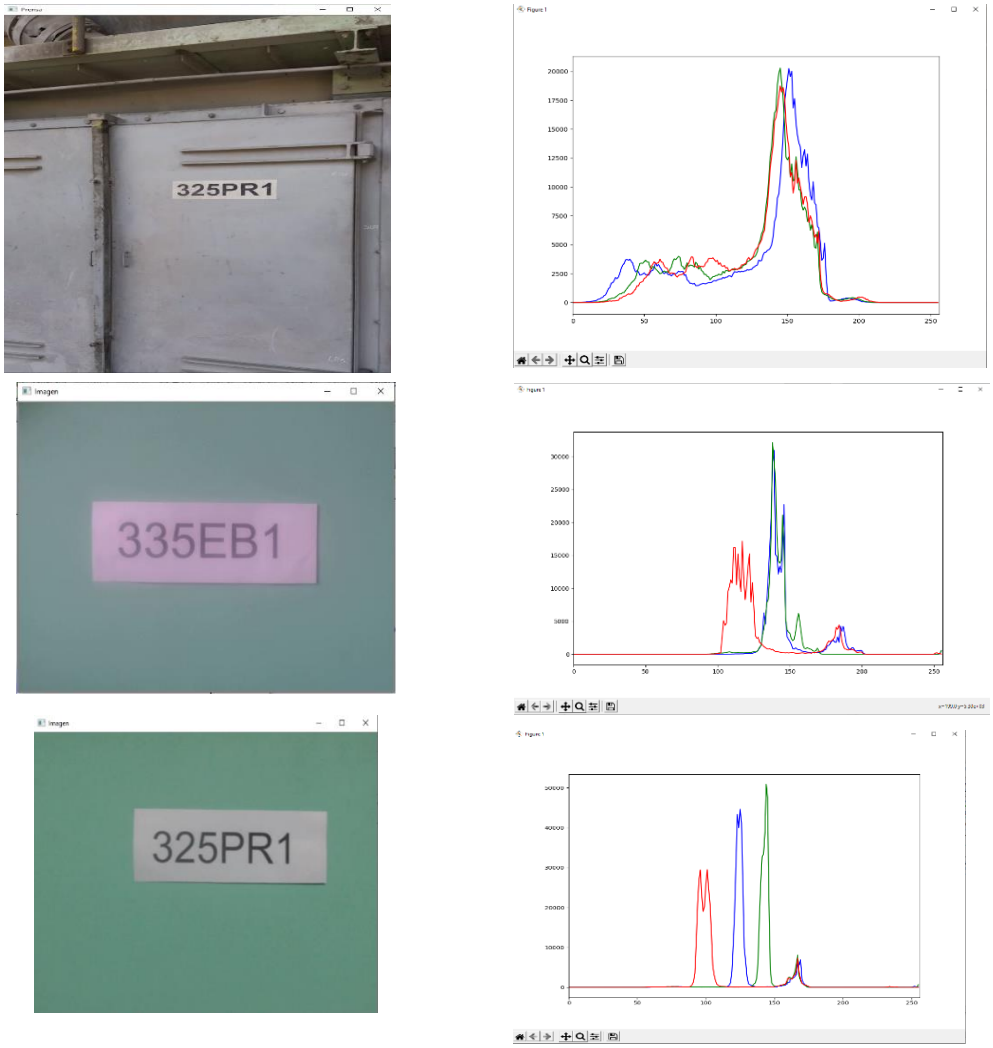
```
import cv2
from matplotlib import pyplot as plt
Imagen=cv2.imread('IMAGEN.jpg')
Colores=('b','g','r')
for i,col in enumerate(Colores):
    Histograma=cv2.calcHist([Imagen],[i],None,[256],[0,256])
    plt.plot(Histograma,color= col)
    plt.xlim([0,256])
plt.show()
```

Fuente: Elaboración Propia

Una vez obtenido el histograma de cada placa se observa que tiene un contraste regular debido que se esparce por todas las intensidades y tiene una excelente iluminación debido

que se encuentran los valores en la parte central, como se muestran en la siguiente figura 38.

Figura 38 Placas identificativas con su respectivo histograma



Fuente: Elaboración Propia

3.8.3 Interfaz de la aplicación

Para la interfaz de la aplicación se necesita la librería Tkinter en el cual se configura el título, los botones, el posicionamiento y tamaño del video. Se utiliza el siguiente código para poder crear una interfaz como se muestra en la figura 39.

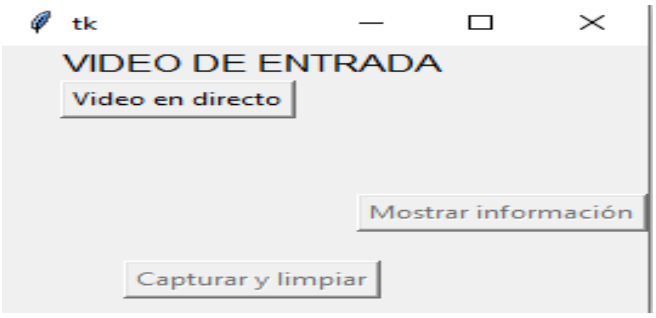
Figura 39: Código para crear la interfaz

```
cap = None
root = Tk()

Titulo = Label(root, text="VIDEO DE ENTRADA", font="bold")
Titulo.grid(column=0, row=0, columnspan=2)

selected = IntVar()

Boton1 = Button(root, text="Video en directo", state="active", command=video_de_entrada)

Boton1.grid(column=0, row=1)

InfVideo = Label(root, text="", width=20)
InfVideo.grid(column=0, row=2)
Video = Label(root)
Video.grid(column=0, row=3, columnspan=2)
Boton2 = Button(root, text="Mostrar información", state="disabled", command=Mostrar)
Boton2.grid(column=1, row=4, columnspan=2, pady=10)
Boton3 = Button(root, text="Capturar y limpiar", state="disabled", command=Capturar_limpiar)
Boton3.grid(column=0, row=5, columnspan=2, pady=10)
root.mainloop()
```

Fuente: Elaboración Propia

En la siguiente figura 40 se observa la interfaz creada al escribir el código anterior.

Figura 40: Interfaz del proyecto



Fuente: Elaboración Propia

Una vez que se creó la interfaz se tiene que poner la funcionalidad que realiza cada botón, por ejemplo, el Boton1 llamado Video en directo llama a la función video_de_entrada que es la encargada de mostrar todo el video como se muestra en la figura 41.

Figura 41: Programación de la interfaz video en directo

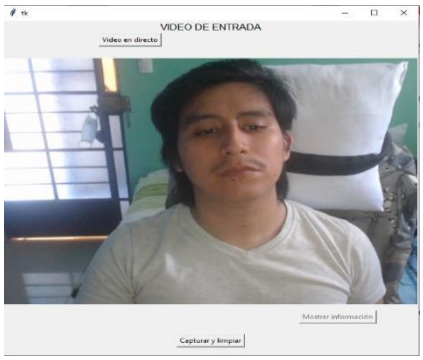
```
def video_de_entrada():
    global cap

    Boton3.configure(state="active")
    Boton2.configure(state="disabled")
    Boton1.configure(state="active")
    InfVideo.configure(text="")
    cap = cv2.VideoCapture(0, cv2.CAP_DSHOW)
    visualizar()
```

```
def visualizar():
    global cap
    global im
    ret, frame = cap.read()
    if ret == True:
        frame = Deteccion_placa(frame)
        frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
        im = Image.fromarray(frame)
        img = ImageTk.PhotoImage(image=im)
        Video.configure(image=img)
        Video.image = img
        Video.after(10, visualizar)


    else:
        Video.image = ""
        InfVideo.configure(text="")
        Boton1.configure(state="active")
        selected.set(0)
        Boton3.configure(state="disabled")
        cap.release()
```

Fuente: Elaboración Propia

Al hacer click al video de entrada se abre el video en directo en el cual detecta y reconoce a las placas y si no se encuentran se muestra la imagen como la figura 42.

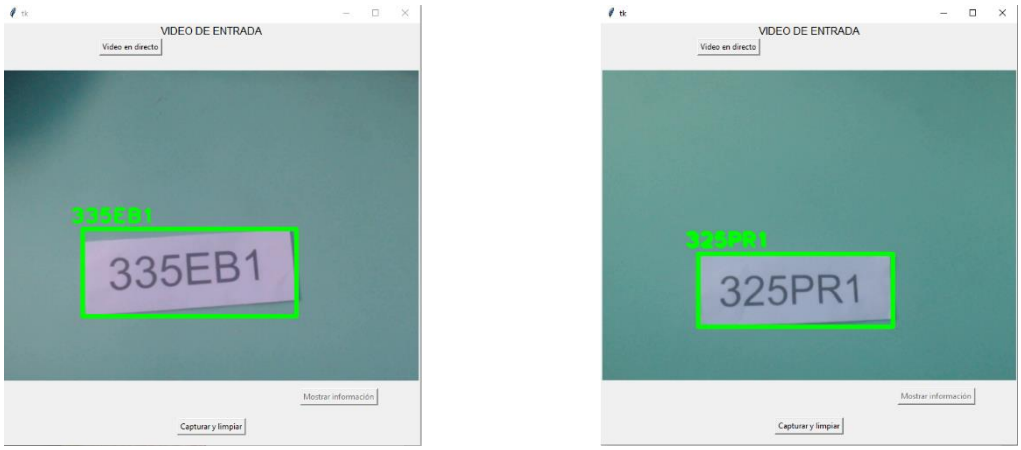
Figura 42: Imagen mostrada por el video en directo



Fuente: Elaboración Propia

En la figura 43 se observa lo que sucede cuando en las imágenes capturadas se muestra una placa identificativa.

Figura 43: Placas identificadas en tiempo real



Fuente: Elaboración Propia

Una vez que se tiene el video en directo se tiene que decidir en la interfaz lo que sucede cuando se presiona el botón Capturar y limpiar en el cual, activa el botón 2 (Mostrar información) y captura la imagen. En la figura 44 se observa el código.

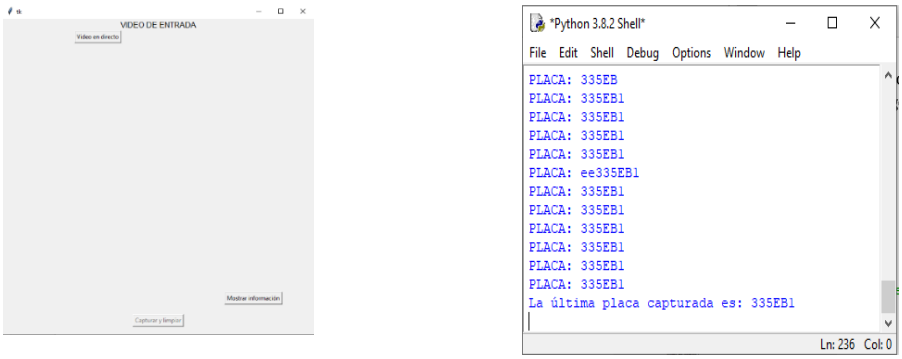
Figura 44: Código de la captura de la imagen

```
def Capturar_limpiar():
    cap.release()
    Video.image = ""
    InfVideo.configure(text="")
    Boton2.configure(state="active")
    Boton1.configure(state="active")
    selected.set(0)
    print('La última placa capturada es:',textoR)
```

Fuente: Elaboración Propia

En la siguiente figura 45 se observa que cuando se presiona el botón (capturar y limpiar) se captura la imagen y se guarda la placa capturada en la variable (Placa) además se limpia el video en directo y se activa el Botón 2 que es Mostrar información.

Figura 45: Captura la imagen y limpia el video en directo



Fuente: Elaboración Propia

Al hacer click en el botón de (Mostrar información) nos da como resultado la última foto tomada con la detección de la placa respectiva, además de darnos información si la placa detectada pertenece a la base de datos guardado. En la figura 46 se observa el código.

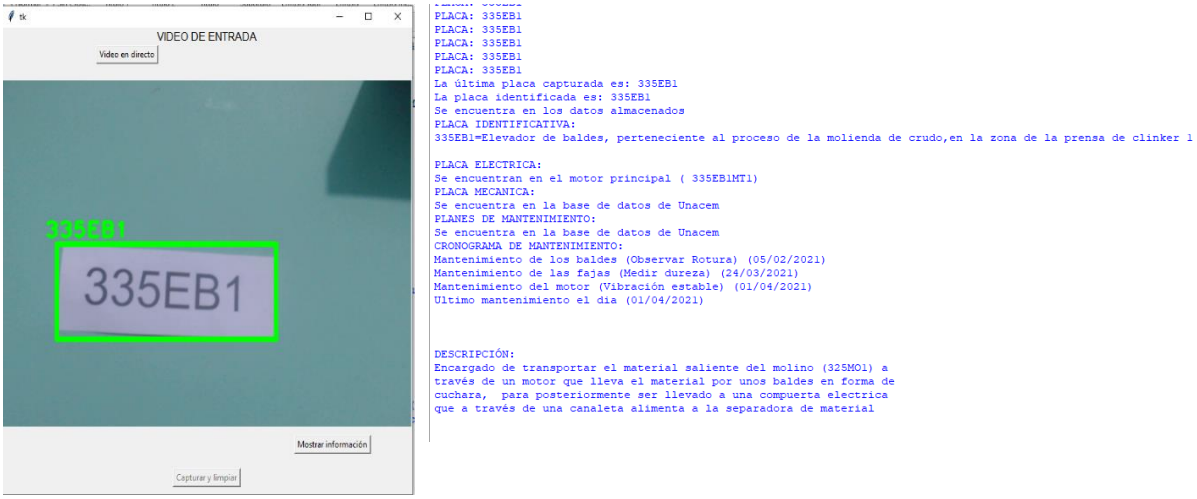
Figura 46: Código para mostrar la información

```
pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-OCR\tesseract'
Excel1= r'D:\PYTHON EJEMPLOS\RECONOCIMIENTO DE PLACAS\Prensa325PR1.xlsx'
Excel2= r'D:\PYTHON EJEMPLOS\RECONOCIMIENTO DE PLACAS\Placa335EB1.xlsx'
def Mostrar():
    global cap
    cap.release()
    img = ImageTk.PhotoImage(image=im)
    Video.configure(image=img)
    Video.image = img
    print('La placa identificada es:',textoR)

    if textoR=='325PR1':
        print('Se encuentra en los datos almacenados')
        AbrirExcel= xlrd.open_workbook(Excel1)
        Hoja=AbrirExcel.sheet_by_name("Hoja1")

        for i in range(Hoja.nrows):
            print(Hoja.cell_value(i,0))

    elif textoR=='335EB1':
            print('Se encuentra en los datos almacenados')
            AbrirExcel= xlrd.open_workbook(Excel2)
            Hoja=AbrirExcel.sheet_by_name("Hoja1")

            for i in range(Hoja.nrows):
                print(Hoja.cell_value(i,0))
    else:
        print('No se encuentra en los datos almacenados')
```

Fuente: Elaboración Propia

En la figura 47 se observa la placa identificativa con su código correcto además que en los resultados nos da la información deseada, debido a que no tengo acceso a la información completa de cada máquina solo introduje lo que conocía.

Figura 47: Placa identificada e información de la máquina



Fuente: Elaboración Propia

3.8.4 Resultados en el procesamiento de la imagen

En la etapa del procesamiento esta encargada de varias sub etapas como las siguientes

3.8.4.1 Convertir la imagen a grises

Para realizar la conversión a escala de grises se utiliza el siguiente código después de adquirir la imagen

```
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
```

En el cual es esencial para poder binarizar la imagen y detectar los bordes, en la siguiente figura 48 se muestra una parte del subproceso Deteccion_placa (frame) encargada de procesar toda la placa y observar que se utiliza el código mencionado para poder convertir a escala de grises una imagen.

Figura 48: Código que convierte la imagen de BGR a escala de grises

```
def Deteccion_placa(frame):
    global text
    global textoR

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
```

Fuente: Elaboración Propia

En la siguiente figura 49 se observan las imágenes con sus conversiones a escala de grises respectivo.

Figura 49: Imágenes de BGR a escala de grises

Fuente: Elaboración Propia

3.8.4.2 Eliminar el ruido de la imagen

Una vez que se convierte a grises se necesita eliminar el mayor ruido posible de una imagen para evitar que no se encierre correctamente la placa identificativa o haya falsos contornos encerrados. Para la reducción del ruido existen varios filtros en el cual probaremos los 2 más utilizados.

3.8.4.2.1 Filtro BLUR

Se utiliza el código blur = cv2.blur(gray, (3,3)) para realizar el filtro y eliminar el ruido, podemos ver su comportamiento respecto a la imagen utilizando un kernel de 3x3 en la

Figura 50: Eliminación de ruido con Filtro Blur

Fuente: Elaboración Propia

figura 50.

3.8.4.2.2 Filtro Gaussiano

Se utiliza el código Gauss = cv2.GaussianBlur(gray, (3,3),0) para realizar el filtro y eliminar el ruido, podemos ver su comportamiento respecto a la imagen utilizando un kernel gaussiano de 3x3 en la figura 51.

Figura 51: Eliminación de ruido con Filtro Gaussiano

Fuente: Elaboración Propia

Se llega a la conclusión que utilizar el Blur que es un filtro Promedio se obtiene mayor disminución del ruido respecto al Gaussiano, por ello en nuestro proyecto se utilizará el filtro Blur.

3.8.4.3 Conversión a Binario

La conversión en binaria de una imagen es utilizada para poder discriminar el fondo de la placa identificativa, por lo cual se utiliza en el procesamiento de la placa en tiempo real. Para poder realizar la binarización se necesita un umbral que me indique de que intensidad es negra o blanca y para obtener el umbral se necesita utilizar métodos en un histograma

para obtener el adecuado. El siguiente código en la figura 52 indica como obtener el histograma de la figura en escala de grises.

Figura 52: Código para obtener el histograma en escala de grises

Prueba Histo Gray.py - D:\PYTHON EJEMPLOS\RECONOCIMIENTO DE PLACAS\HISTOGRAMA.

File Edit Format Run Options Window Help

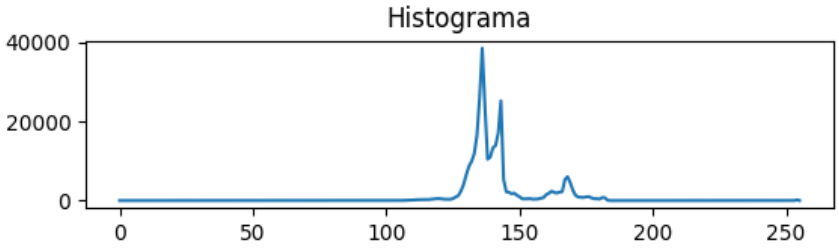
```
import cv2
from matplotlib import pyplot as plt
Imagen=cv2.imread('IMAGEN.jpg')
Gray=cv2.cvtColor(Imagen, cv2.COLOR_RGB2GRAY)
Histo=cv2.calcHist([Gray],[0],None,[256],[0,255])
Figura=plt.figure(1)
Plot1=Figura.add_subplot()
Plot1.plot(Histo)
plt.title('Histograma')
Figura.show()
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Fuente: Elaboración Propia

En la siguiente figura 53 se observa que el histograma de la placa identificativa en fondo verde (335EB1) es bimodal por el cual se puede diferenciar el fondo de la placa, además en el histograma de la placa se observa que después del valor de 150 es la placa identificativa y antes es considerado el fondo que va desde 120 a 150 que sería de color verde, las letras en negro están por los valores de 0 a 100. Por ello se utiliza el método de valor mínimo local observado en el capítulo 2 para obtener el umbral, por lo tanto, el umbral es 150.

Utilizando el código _, binarizada= cv2.threshold(gray,150, 255, cv2.THRESH_BINARY)

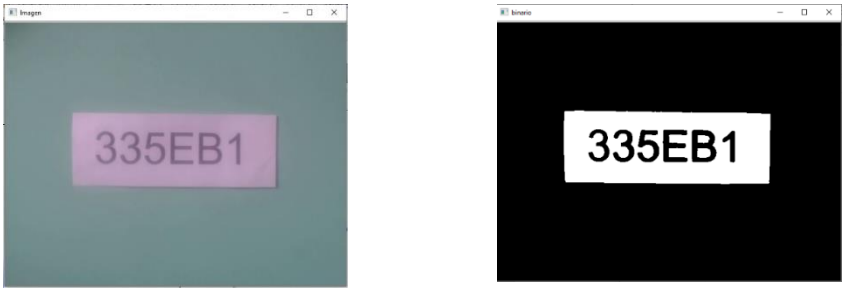
Figura 53: Histograma en grises de la placa en fondo verde (335EB1)



Fuente: Elaboración Propia

Se obtiene la binarización de la imagen, observando en la figura 54 que usando el valor mínimo del umbral se da de manera correcta y no se necesita utilizar otros métodos.

Figura 54: Binarización de la placa en fondo verde (335EB1)

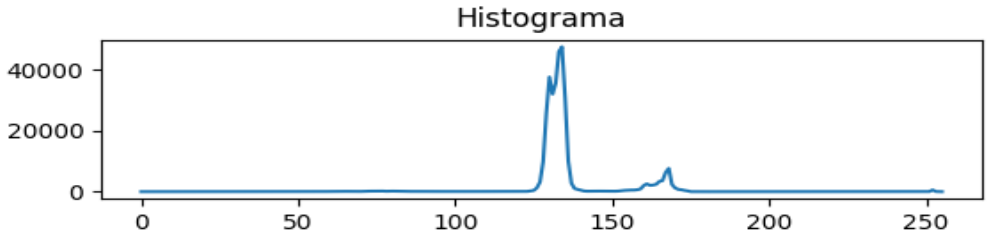


Fuente: Elaboración Propia

Ahora se realiza el histograma de la placa en fondo verde (325PR1) para observar el histograma y saber si cumple el mismo valor de umbralización, se observa en la figura 55

que al poner el mismo umbral de 150 no afecta debido que es el mismo color de fondo, y se puede sacar la conclusión de igual manera que de 150 para arriba es la placa, de 120 a 150 es el fondo verde y antes de 100 son las letras negras.

Figura 55: Histograma en grises de la placa en fondo verde (325PR1)



Fuente: Elaboración Propia

Utilizando el mismo código

_,binarizada=cv2.threshold(gray,150, 255, cv2.THRESH_BINARY)

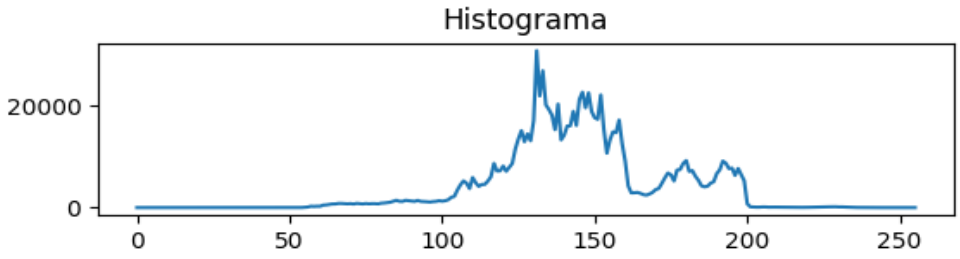
Se obtiene la binarización de la imagen, además se observa en la figura 56 que usando el valor mínimo del umbral se da de manera correcta y no se necesita utilizar otros métodos.

Figura 56: Binarización de la placa en fondo verde (325PR1)

Fuente: Elaboración Propia

Ahora el método de binarizar no es eficiente en las imágenes de la placa identificativa de la fábrica debido a los cambios de umbrales en cada imagen porque el fondo es muy variante, el histograma tiene varios valores de intensidad y muestra que no es bimodal por lo tanto no se puede diferenciar la placa identificativa del fondo. En la siguiente figura 57 se observa el histograma de la placa 325PR1 observando que no es bimodal debido que no se puede diferenciar el fondo de la placa.

Figura 57: Histograma en grises de la placa de la fábrica (335EB1)



Fuente: Elaboración Propia

Por lo tanto, la mejor manera de obtener el umbral es ir probando valores que reconozca únicamente a la placa identificativa debido que otros métodos utilizados nos dan valores que no encierran a la placa identificativa. En este caso conseguimos que la mejor manera de obtener es con un valor de umbral de 160, incluso se prueba con el método de otsu dando una imagen que no encierra el borde de la placa, como se muestra en la figura 58.

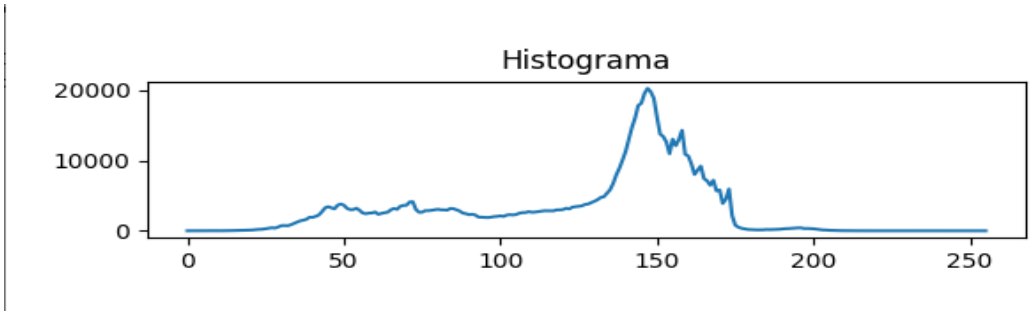
Figura 58: Binarización de la placa de la fábrica utilizando el umbral de 160 y el método de Otsu (335EB1)

(Imagen original) (Binarizado con umbral 160) (Binarizado con método de Otsu)

Fuente: Elaboración Propia

Una vez que se obtuvo el umbral se prueba con otra imagen tomado en la fábrica, primero veremos que el histograma de igual manera es no bimodal por lo cual no se puede diferenciar a simple vista el fondo de la placa, se muestra en la figura 59.

Figura 59: Histograma en grises de la placa de la fábrica (325PR1)



Fuente: Elaboración Propia

Al utilizar el mismo umbral de 160 se observa en la figura 60 que la placa no se encierra siendo una técnica no eficiente cuando se toma con varios objetos alrededor, normalmente

este paso se obvia y se va de frente a la detección de bordes. incluso se prueba con el método de otsu dando una imagen limpia, pero sin encerrar el borde esencial de la placa.

Figura 60: Binarización de la placa de la fábrica utilizando el umbral de 160 y el método de Otsu (325PR1)

(Imagen original) (Binarizado con umbral 160) (Binarizado con método de Otsu)

Fuente: Elaboración Propia

La conclusión que podemos sacar es que este método es obviado para imágenes obtenidos en la fábrica debido a los diferentes objetos, pero se podría realizar si es que la captura de imagen en la fábrica sería mostrando únicamente la placa con un fondo uniforme como las placas identificativas en fondo verde procesadas en tiempo real.

3.8.4.4 Detección de bordes (Canny)

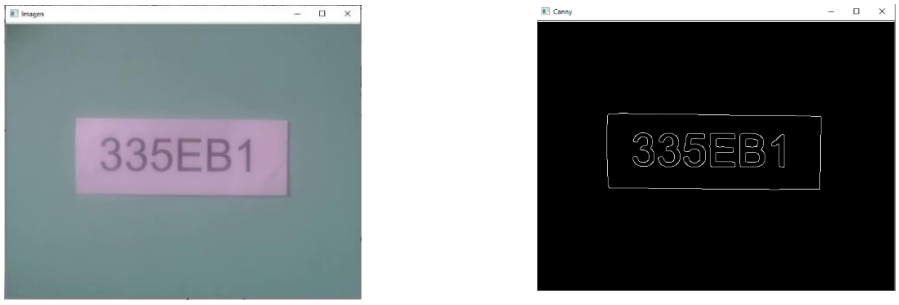
Para detectar los bordes existen varios métodos, el cual el más utilizado es el algoritmo Canny debido a sus etapas y su eficiencia, por lo cual se empleará y se verá que tan bien detecta los bordes de la placa. El algoritmo canny se expresa en el siguiente código.

```
canny = cv2.Canny(binarizada,100,150)
```

el cual, la primera es la variable que se procesa, la segunda y tercera son el valor mínimo y valor máximo del umbral que deciden cuáles serán los bordes, para conseguir los valores más eficientes se empieza a probar valores con la imagen hasta que nos de la placa identificativa correctamente encerrada.

En la siguiente figura 61 se observa la utilidad del algoritmo canny pasando anteriormente la imagen por la binarización, debido que se pasó por esa etapa, los valores mínimos y máximos son fáciles de obtener porque el contorno está encerrado en blanco y negro, en este caso se toma los valores 100, 150 como minino y máximo valor del umbral.

Figura 61: Detección de bordes Canny en placa en fondo verde (335EB1)



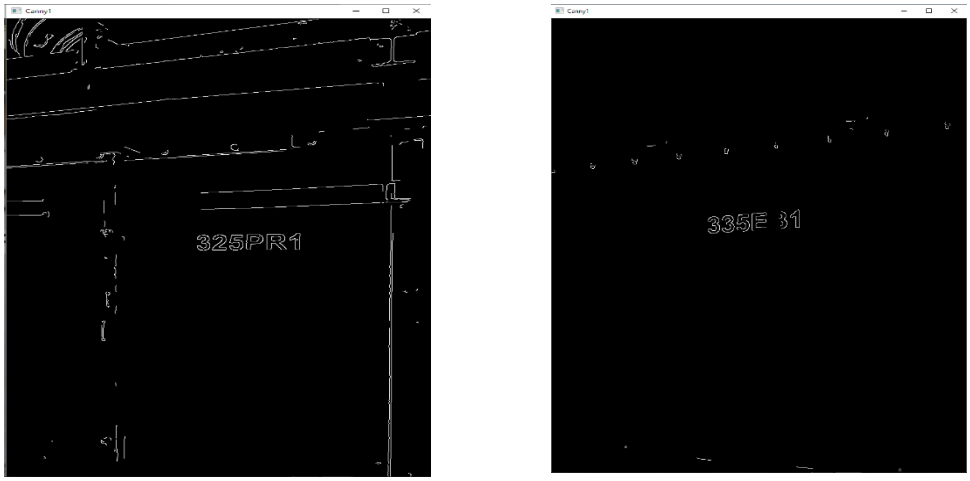
Fuente: Elaboración Propia

Si la imagen no es pasada anteriormente por la etapa de binarización, como el caso de las placas identificativas de la fábrica, se necesita probar varias veces debido que el contorno no es rápidamente encontrado, se muestran las imágenes procesadas con distintos valores

.

En la siguiente figura 62 de las placas de la fábrica 325PR1 y 335EB1 se muestra el resultado de utilizar el canny con el umbral 150 y 200, se observa que no se encierra la placa identificativa. El código es (canny = cv2.Canny(gray,150,200)).

Figura 62: Canny con umbral (150, 200) en placas de la fábrica (325PR1 y 335EB1)

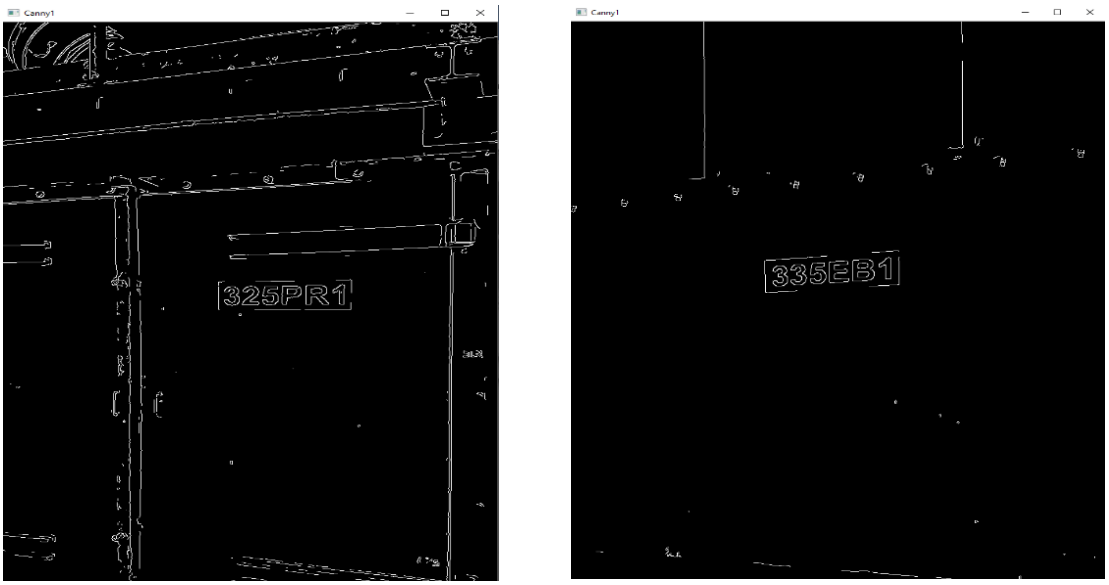


Fuente: Elaboración Propia

En la siguiente figura 63 de las placas de la fábrica 325PR1 y 335EB1 se muestra el resultado de utilizar el canny con el umbral 100 y 120, se observa que se ha mejorado la detección de la placa, pero aún no está encerrado correctamente habiendo algunos agujeros además que mientras menos sea el valor más borde se presentan en las placas.

canny = cv2.Canny(gray,100,120)

Figura 63: Canny con umbral (100, 120) en placas de la fábrica (325PR1 y 335EB1)

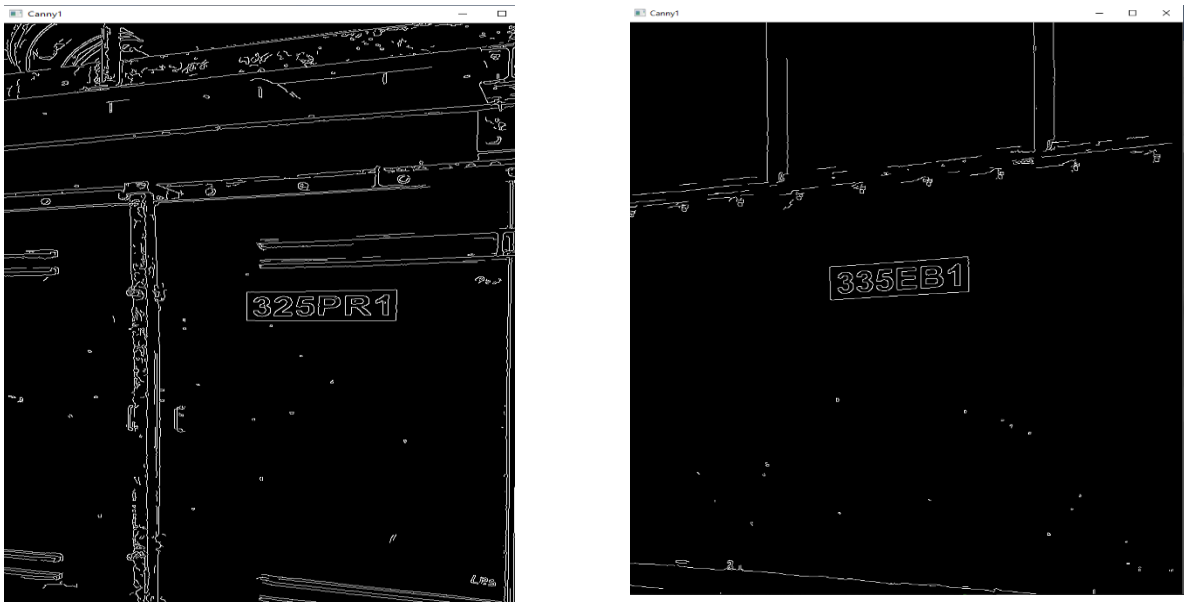


Fuente: Elaboración Propia

En la siguiente figura 64 de las placas de la fábrica 325PR1 y 335EB1 se muestra el resultado de utilizar el canny con el umbral 50 y 80, se observa que la placa se ha encerrado correctamente pudiendo ser detectada los bordes en la siguiente etapa, además de que se observa que se presentan más bordes.

canny = cv2.Canny(gray,50,80)

Figura 64: Canny con umbral (50, 80) en placas de la fábrica (325PR1 y 335EB1)



Fuente: Elaboración Propia

Se llega a la conclusión que mientras menos sea el valor min y máximo del umbral se detectará más bordes, pudiendo procesar la imagen para obtener el código de la placa, en este caso hemos usado el umbral min 50 y max 80 pero habrá casos en máquinas que se necesite menos o más dependiendo de la iluminación o bordes que lo rodeen, por ello lo más recomendable es tomar únicamente la placa con un fondo fijo para poder realizar primero la binarización y pasarlo luego por el canny, y así obtener valores fijos en cualquier lugar de la fábrica.

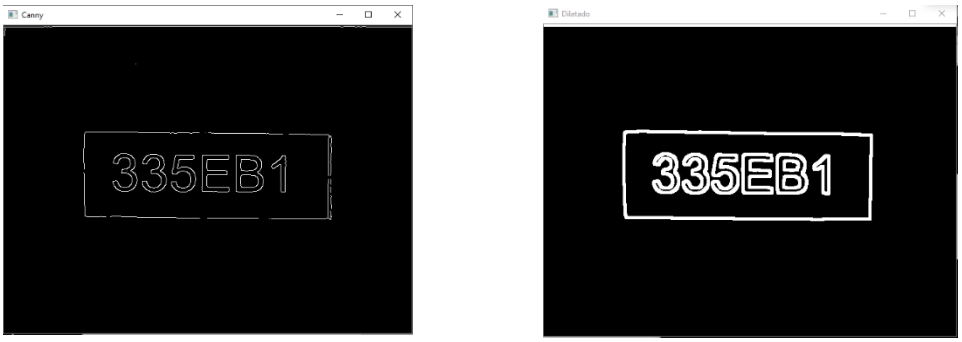
3.8.4.5 Operaciones morfológicas

En las operaciones morfológicas se utiliza el dilate para poder aumentar el grosor de las detecciones de los bordes y así evitar contornos no encerrados que pueden contener a la placa identificativa, se utiliza el siguiente código.

```
Dilate = cv2.dilate(canny, None, iterations=1)
```

En la siguiente figura 65 se muestra que el rectángulo no encierra a la placa identificativa dando como resultado que el programa no pueda reconocer a la placa identificativa, pero al usar dilate en la imagen el rectángulo queda encerrado, dando como resultado una imagen que se puede reconocer.

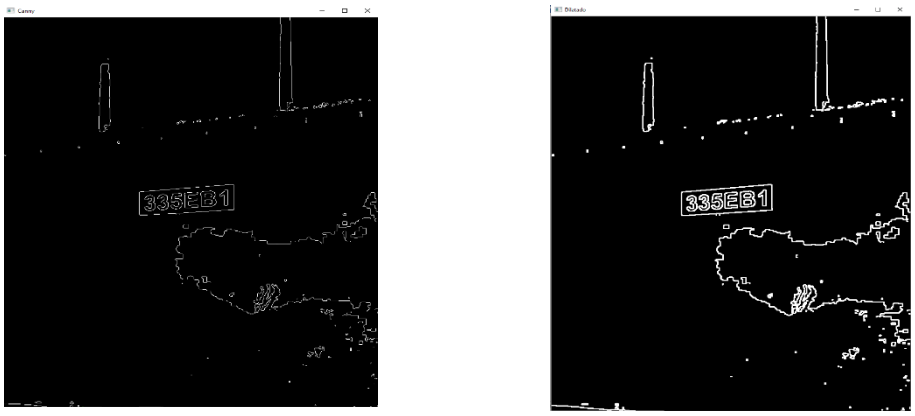
Figura 65: Dilatación en placa con fondo verde (335EB1)



Fuente: Elaboración Propia

En la siguiente figura 66 se muestra como afecta el dilate en la imagen de la fábrica Unacem.

Figura 66: Dilatación en placa de la fábrica (335EB1)



Fuente: Elaboración Propia

3.8.4.6 Encontrar y dibujar los contornos

Se utiliza el siguiente código mostrado a continuación en el cual RETR_LIST equivale a todos los contornos de la imagen y se muestra los contornos con drawContours.

```
Contornos,_=cv2.findContours(canny,cv2.RETR_LIST,cv2.CHAIN_APPROX_SIMPLE)

cv2.drawContours(image, Contornos -1, (0,255,0),2)
```

Esto sirve principalmente para encontrar todos los contornos y guardarlo en una variable que en este caso es llamado Contornos, y poder discriminar el área de cada contorno como se mostrará en la figura 67.

Figura 67: Contornos en las placas identificativas (325PR1)

Fuente: Elaboración Propia

3.8.4.7 Reconocer el contorno deseado

En esta etapa se discrimina el área de cada contorno encerrado, teniendo en cuenta la variable (Contornos) obtenido del proceso anterior. Así se evita que cualquier contorno pueda ser detectado como la placa identificativa, se realiza el código de la figura 68 para discriminar los contornos que no se parezcan a un rectángulo debido que nuestra placa identificativa tiene esa forma.

Figura 68: Código para reconocer la placa de los demás contornos

```
for c in Contornos:
  Area = cv2.contourArea(c)
  X,Y,W,H = cv2.boundingRect(c)
  Epsilon = 0.09*cv2.arcLength(c,True)
  AproxRect = cv2.approxPolyDP(c,Epsilon,True)
  if len(AproxRect)==4 and Area>10000:
    Aspecto = float(W)/H

    if Aspecto>2.4:
      Placa= gray[Y:Y+H,X:X+W]
```

Fuente: Elaboración Propia

En el código se observa que la variable c equivale a los contornos encontrados y se va discriminando 1 por 1 hasta llegar a la variable (Contornos), las variables (Epsilon) y (AproxRect) son usados para diferenciar el rectángulo.

La variable (Aspecto) es el Aspect Ratio de la medida del rectángulo que debe de cumplir para que sea detectado como placa identificativa, para hallarlo se divide el ancho entre la altura. Se pone luego que el (Aspecto) de los rectángulos deben de cumplir con ser mayor a 2.4 para que sean identificados como placa identificativa debido que nuestra placa tiene un valor de (Aspecto) igual a 2.7. cómo se observa en la figura 69.

Figura 69: Longitud de la placa y su Aspect Ratio

Aspecto=Aspect_Ratio= (175mm) / (65mm) = 2.7

Fuente: Elaboración Propia

En la siguiente figura 70 se muestra la variable placa que contiene únicamente a la placa identificada de toda la imagen.

Figura 70: Imagen de la variable (placa) que contiene la placa identificativa

Fuente: Elaboración Propia

3.8.4.8 Reconocer y mostrar los caracteres de la placa

Para el sistema de reconocimiento de caracteres se utiliza la librería pytesseract debido que permite utilizar Tesseract en Python que es uno de los OCR con más precisión al reconocer caracteres, está encargada específicamente de descifrar cada carácter de la placa encerrara, la placa encerrada se encuentra en la variable (placa), la variable configuración sirve para elegir el modo de segmentación que se va a utilizar, en este caso se elige el 11 debido el reconocimiento de caracteres busca la mayor cantidad de texto posible en la placa identificativa sin ningún orden en particular como el siguiente código.

Configuracion=r'- -psm 11'
text=pytesseract.image to string (Placa, config=configuracion)

Una vez que se obtiene el texto de la placa identificativa es guardado en la variable text, se muestra con el cv2.rectangle el contorno de la placa detectada y con el cv2.putText muestra el texto encima de la placa en tiempo real, se muestra a continuación con el siguiente código en la figura 71.

Figura 71: Código para reconocer los caracteres de una placa en un video en directo

```
Configuracion= r'--psm 11'
text= pytesseract.image_to_string(Placa,config=Configuracion)
frame= cv2.rectangle(frame,(X,Y),(X+W,Y+H),(0,255,0),3)
cv2.putText(frame,textoR,(X-20,Y-10),1,2.2,(0,255,0),3,cv2.LINE_AA)
```

Fuente: Elaboración Propia

En la siguiente figura 72 se observa la detección en tiempo real de cada imagen del video en directo.

Figura 72: Placa identificada en tiempo real del video en directo

Fuente: Elaboración Propia

El código siguiente mostrado en la figura 73 es el usado para procesar una imagen de la fábrica Unacem, es parecida que la anterior, pero con la diferencia que en esta ocasión solo se observa una imagen por ello la variable es image y no frame.

Figura 73: Código para reconocer los caracteres de una placa en una imagen

```
configuracion= r'--psm 11'
text= pytesseract.image_to_string(placa,config=configuracion)
cv2.rectangle(image,(x,y),(x+w,y+h),(0,255,0),3)
cv2.putText(image,text,(x-20,y-10),1,2.2,(0,255,0),3)
```

Fuente: Elaboración Propia

En la siguiente figura 74 se observan las imágenes con su respectivo código identificativo.

Figura 74: Placa identificada de las imágenes de la fábrica

Fuente: Elaboración Propia

3.8.4.9 Mostrar la última placa capturada

Al presionar captura y limpiar en la interfaz se debe guardar la última placa capturada para poder mostrarla, pero si por el movimiento de la cámara o el ruido el sistema de reconocimiento no captura nada, la variable texto se guarda como si estuviera vació y provoca que el texto este en blanco, para evitar esto se realiza el siguiente código para mostrar siempre el último código detectado cuando se presione el botón de mostrar información. Como se observa en el código de la figura 75 se escribe una variable llamada textoR que es guardado siempre y cuando haya algún código en la variable text.

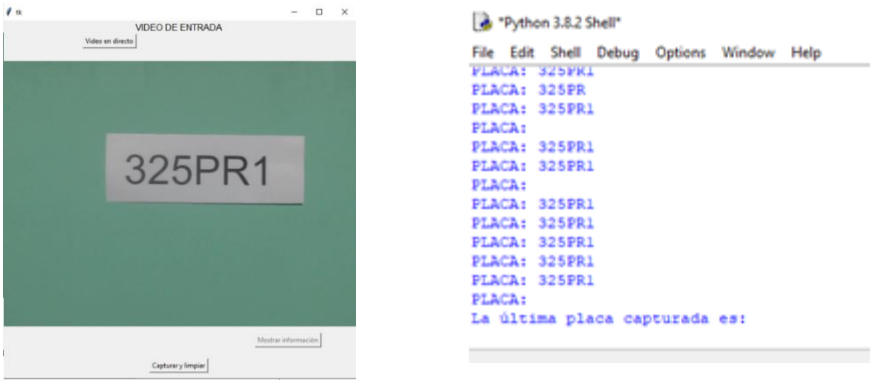
Figura 75: Código para mostrar la última placa capturada

```
            if not text=='':
                textoR=text
                print('PLACA:',textoR)
                frame= cv2.rectangle(frame,(X,Y),(X+W,Y+H),(0,255,0),3)
                cv2.putText(frame,textoR,(X-20,Y-10),1,2.2,(0,255,0),3,cv2.LINE_AA)
            else:
                frame= cv2.rectangle(frame,(X,Y),(X+W,Y+H),(0,255,0),3)
                cv2.putText(frame,text,(X-20,Y-10),1,2.2,(0,255,0),3,cv2.LINE_AA)
        return frame


def Capturar_limpiar():
    cap.release()
    Video.image = ""
    InfVideo.configure(text="")
    Boton2.configure(state="active")
    Boton1.configure(state="active")
    selected.set(0)
    print('La última placa capturada es:',textoR)
```

Fuente: Elaboración Propia

En la siguiente figura 76 se observa que si no se realiza un cambio en la variable (text) se puede mostrar una imagen sin procesar.

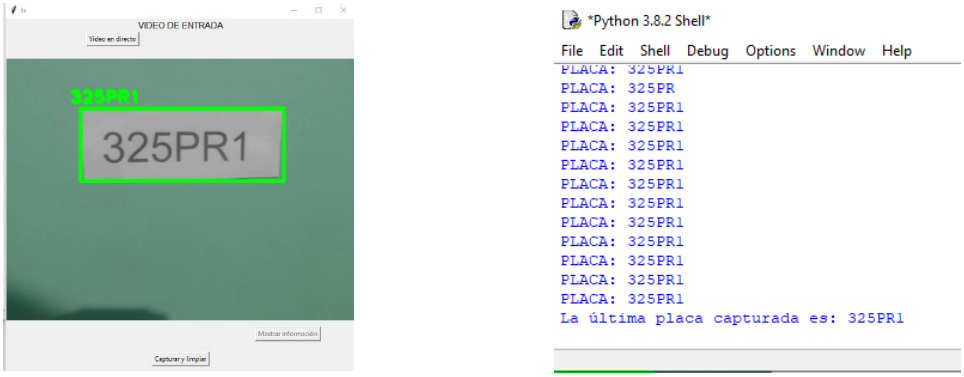
Figura 76: Placa mostrada sin procesar debido al ruido o desenfoque (325PR1)



Fuente: Elaboración Propia

En la siguiente figura 77 se observa que gracias al cambio de variable (textoR) siempre se muestra un valor de la placa por lo cual cuando se seleccione mostrar información me dará una imagen con la última placa detectada y no una imagen sin procesar.

Figura 77: Placa identificada correctamente en fondo verde (325PR1)



Fuente: Elaboración Propia

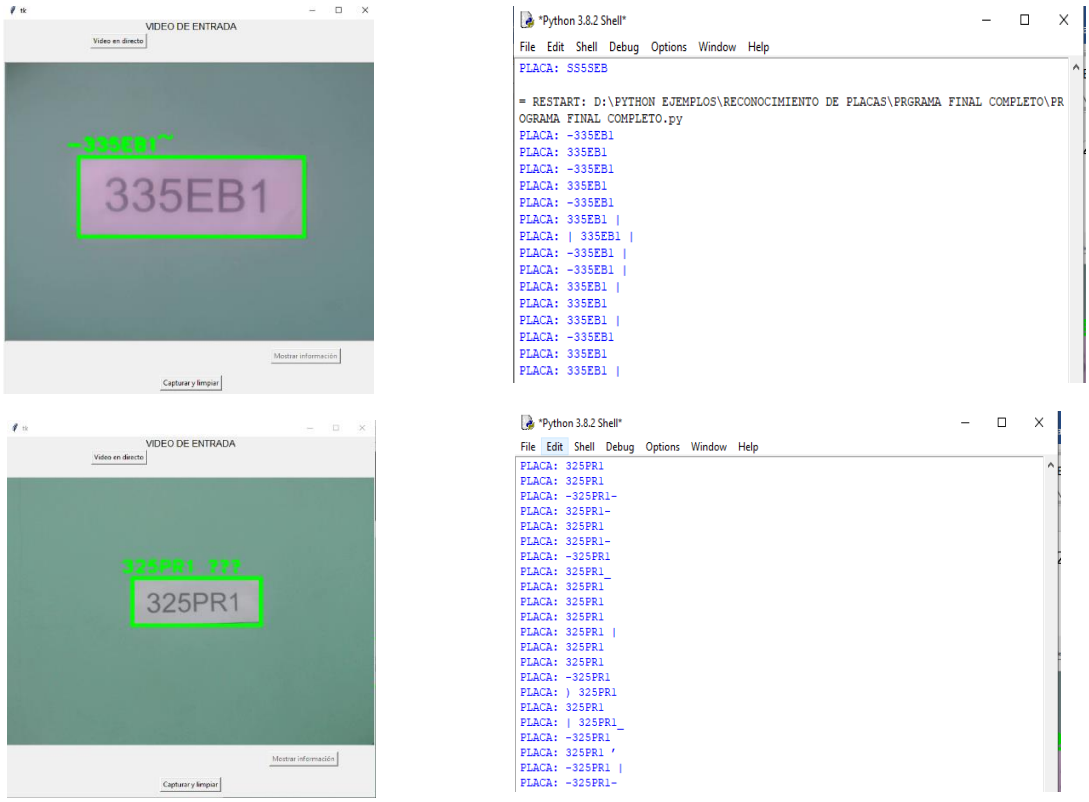
3.8.4.10 Eliminar los caracteres especiales y minúsculas

Para eliminar los caracteres especiales y minúsculas se utiliza el siguiente código

```
Patron='[a - z]'
textoR = ' '. join(filter(str.isalnum, textoR))
textoR=re.sub(Patron,' ',textoR)
```

Con este código se elimina todos los caracteres especiales como la arroba, paréntesis, signo de pregunta, etc. Además, se eliminan las minúsculas que se muestran en una imagen debido que el detector de caracteres toma algunos ruidos o borrosidad de la imagen al moverse en tiempo real como si perteneciera a algún carácter. En la siguiente figura 78 se observan los resultados obtenidos sin un filtro de caracteres especiales y minúsculas.

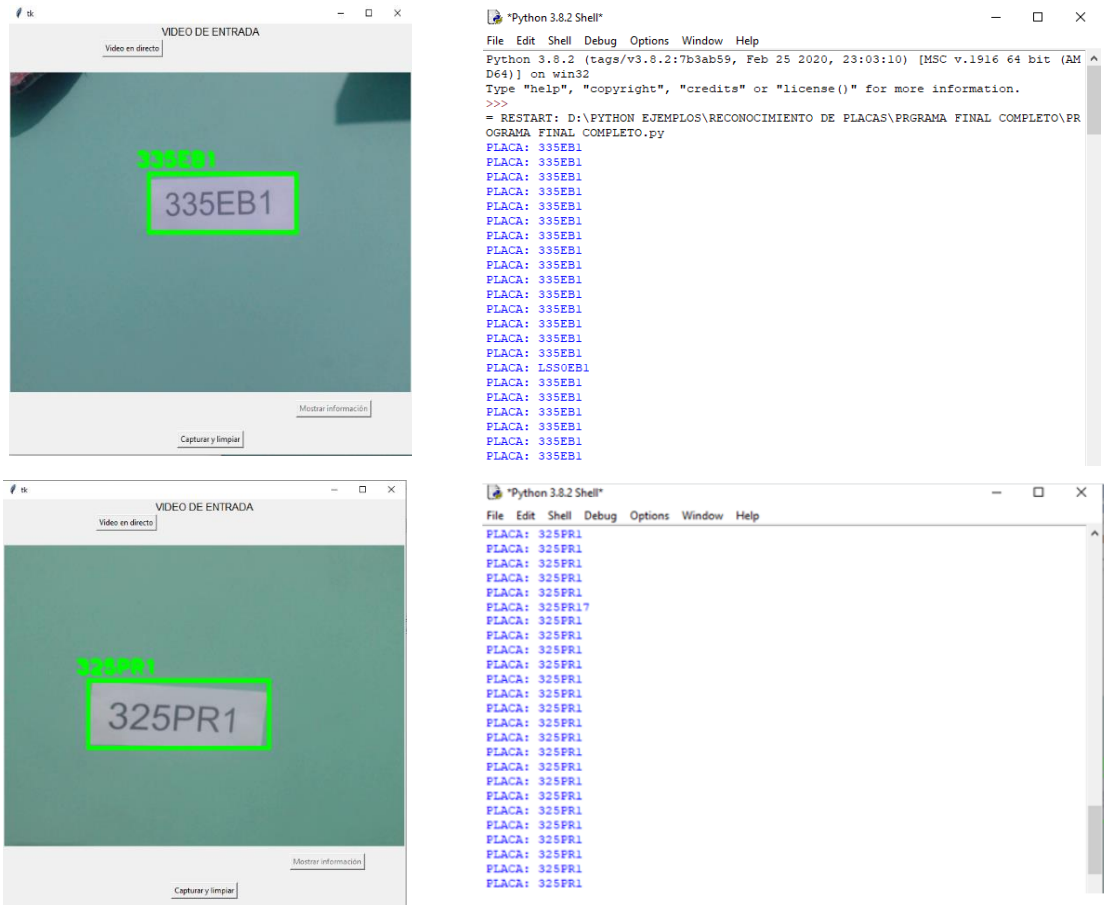
Figura 78: Caracteres especiales que no pertenecen a la placa



Fuente: Elaboración Propia

Se puede observar que algunos códigos cumplen, pero hay muchos que no y esto es perjudicial a la hora de intentar obtener el código correcto de la placa, por ello al usar un filtro de caracteres especiales y minúsculas, se observa un mejoramiento en la detección de los caracteres en tiempo real, como se muestra en la siguiente figura 79.

Figura 79: Placa capturada después de filtrar los caracteres especiales y minúsculas (335EB1)



Fuente: Elaboración Propia

3.8.5 Mostrar información de la máquina

Para mostrar información de los datos almacenados, se necesita tener archivos en Excel anteriormente guardados con toda la información que se desea saber de una máquina para que cuando sea procesado la imagen se pueda mostrar la información del Excel. Se observa el código en la figura 80.

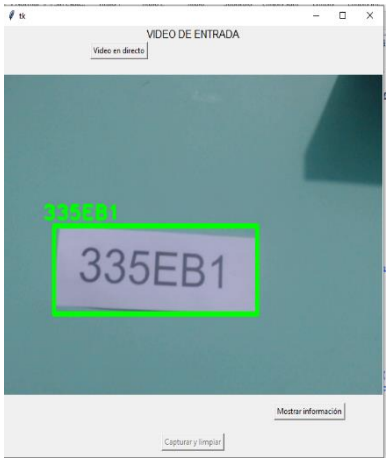
Figura 80: Código para obtener información de la placa que está en la base de datos

```
pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-OCR\tesseract'
Excel1= r'D:\PYTHON EJEMPLOS\RECONOCIMIENTO DE PLACAS\Prensa325PR1.xlsx'
Excel2= r'D:\PYTHON EJEMPLOS\RECONOCIMIENTO DE PLACAS\Placa335EB1.xlsx'
def Mostrar():
    global cap
    cap.release()
    img = ImageTk.PhotoImage(image=im)
    Video.configure(image=img)
    Video.image = img
    print('La placa identificada es:',textoR)

    if textoR=='325PR1':
        print('Se encuentra en los datos almacenados')
        AbrirExcel= xlrd.open_workbook(Excel1)
        Hoja=AbrirExcel.sheet_by_name("Hoja1")

        for i in range(Hoja.nrows):
            print(Hoja.cell_value(i,0))

    elif textoR=='335EB1':
            print('Se encuentra en los datos almacenados')
            AbrirExcel= xlrd.open_workbook(Excel2)
            Hoja=AbrirExcel.sheet_by_name("Hoja1")

            for i in range(Hoja.nrows):
                print(Hoja.cell_value(i,0))
    else:
        print('No se encuentra en los datos almacenados')
```

Fuente: Elaboración Propia

En la figura 81 se observa la placa identificativa 335EB1 con su código correcto además que en los resultados nos da la información deseada, debido a que no tengo acceso a la información de cada máquina solo introduje lo que conocía.

Figura 81: Información completa de la placa en fondo verde (335EB1)




```
PLACA: 335EB1
PLACA: 335EB1
PLACA: 335EB1
PLACA: 335EB1
PLACA: 335EB1
La última placa capturada es: 335EB1
La placa identificada es: 335EB1
Se encuentra en los datos almacenados
PLACA IDENTIFICATIVA:
335EB1=Elevador de baldes, perteneciente al proceso de la molienda de crudo,en la zona de la prensa de clinker 1

PLACA ELECTRICA:
Se encuentran en el motor principal ( 335EB1MT1)
PLACA MECANICA:
Se encuentra en la base de datos de Unacem
PLANES DE MANTENIMIENTO:
Se encuentra en la base de datos de Unacem
CRONOGRAMA DE MANTENIMIENTO:
Mantenimiento de los baldes (Observar Rotura) (05/02/2021)
Mantenimiento de las fajas (Medir dureza) (24/03/2021)
Mantenimiento del motor (Vibración estable) (01/04/2021)
Ultimo mantenimiento el dia (01/04/2021)



DESCRIPCIÓN:
Encargado de transportar el material saliente del molino (325MO1) a
través de un motor que lleva el material por unos baldes en forma de
cuchara,  para posteriormente ser llevado a una compuerta electrica
que a través de una canaleta alimenta a la separadora de material
```

Fuente: Elaboración Propia

CAPITULO 4

RESULTADOS

En el capítulo 4 se observa el resultado de cada etapa de los procesos explicados en el capítulo 3

4.1 Resultados del Pre-procesamiento:

4.1.1 Pre-procesamiento en placa con fondo uniforme

- En el pre-procesamiento se encuentran las operaciones realizadas en la imagen tomada en tiempo real, la imagen es pasada a escala de grises para poder ser binarizada con el código

  gray=cv2.cvtColor(frame.cv2.COLOR_BGR2GRAY)

- Luego de pasarlo por escala de grises, se filtra el ruido de la imagen para eliminar los falsos contornos, el cual se comparó el filtro Blur del Gaussiano y se escoge el filtro Blur que es un filtro promedio con un kernel 3x3

  blur = cv2.blur(gray, (3,3))

- Una vez eliminado el ruido se pasa a binarizar la imagen, el umbral se obtiene viendo el histograma de la imagen, con el valor mínimo local obtenemos que el mejor umbral es el 150 por ello se toma ese valor para poder binarizarlo, se utiliza el código

  _, binarizada= cv2.threshold(gray,150, 255, cv2.THRESH_BINARY)

- Se pasa por un detector de bordes llamado Canny que detecta los bordes de la placa a través de un algoritmo, se elige este método debido a su eficiencia en la detección de bordes, debido que la imagen es pasada por la binarización, el umbral min y max que encierre a la placa es fácil de conseguir, se obtiene que los umbrales son 100 y 150. Se utiliza el siguiente código

canny = cv2.Canny(binarizada,100,150)

- Se utiliza la operación morfológica dilate para poder aumentar el grosor de los contornos y así evitar que el rectángulo de la placa identificativa no se encierre correctamente, se aumenta únicamente 1 grosor de la placa con el siguiente código

Dilate = cv2.dilate(canny, None, iterations=1

Figura 82: Resultados del Pre-procesamiento (335EB1)

Fuente: Elaboración Propia

4.1.2 Pre-procesamiento en placa de la fábrica

- La imagen se pasa por escala de grises (cv2.cvtColor) y se elimina el ruido con el filtro Blur (blur = cv2.blur(gray, (3,3))) de igual manera que la placa con fondo uniforme
- Se obvia la etapa de binarización, debido a que el fondo es muy variante que provoca los cambios de umbral en cada imagen, además como el histograma que se obtiene no es bimodal no se puede diferenciar entre la placa identificativa y el fondo además si se utiliza el método de Otsu para binarizar, se obtiene una binarización que no encierra a la placa identificativa
- Se utiliza el algoritmo Canny para detector los bordes de la placa, pero se necesita ir probando valores para obtener el umbral min y max, se llega a la conclusión que la mejor detección es con el umbral min 50 y max 80
- Se utiliza la dilatación para que el contorno de la placa identificativa se encierre correctamente

Figura 83: Resultados del Preprocesamiento (325PR1)

Fuente: Elaboración Propia

4.1.3 Resultados del reconocimiento

- Para el reconocimiento necesitamos encontrar y dibujar los contornos de la placa para observar si de todos los contornos encerrados se encuentra la placa identificativa, los contornos de la placa se almacenan en la variable (Contornos) con el siguiente código

  Contornos,_=cv2.findContours(canny,cv2.RETR_LIST,cv2.CHAIN_APPROX_SIMPLE)

  cv2.drawContours(image, Contornos -1, (0,255,0),2)

- Luego de tener los contornos encerrados en la variable (Contornos) se discrimina la placa identificativa de todos los contornos encerrados, identificando todos los rectángulos el cual uno de ellos pertenece a la placa, con el siguiente código

```
for c in Contornos:
  Area = cv2.contourArea(c)
  X,Y,W,H = cv2.boundingRect(c)
  Epsilon = 0.09*cv2.arcLength(c,True)
  AproxRect = cv2.approxPolyDP(c,Epsilon,True)
```

- Luego de obtener todos los rectángulos identificados, se discrimina segun su área, se observa que la placa identificativa tiene un área mayor a 10000 pixeles, también se diferencia el tamaño del rectángulo según su aspect Ratio que equivale al ancho dividido con la altura de la placa que debe ser mayor a 2.4

```
if len(AproxRect)==4 and Area>10000:
  Aspecto = float(W)/H

  if Aspecto>2.4:
    Placa= gray[Y:Y+H,X:X+W]
```

- Una véz obtenido cual de todos los contornos es la placa identificativa, se guarda en la variable (Placa) para poder utilizar el reconocimiento de caracteres con Tesseract el cual busca la mayor cantidad de texto posible en la placa identificativa sin ningún orden en particular

```
Configuracion= r'--psm 11'
text= pytesseract.image_to_string(Placa,config=Configuracion)
frame= cv2.rectangle(frame,(X,Y),(X+W,Y+H),(0,255,0),3)
cv2.putText(frame,textoR,(X-20,Y-10),1,2.2,(0,255,0),3,cv2.LINE_AA)
```

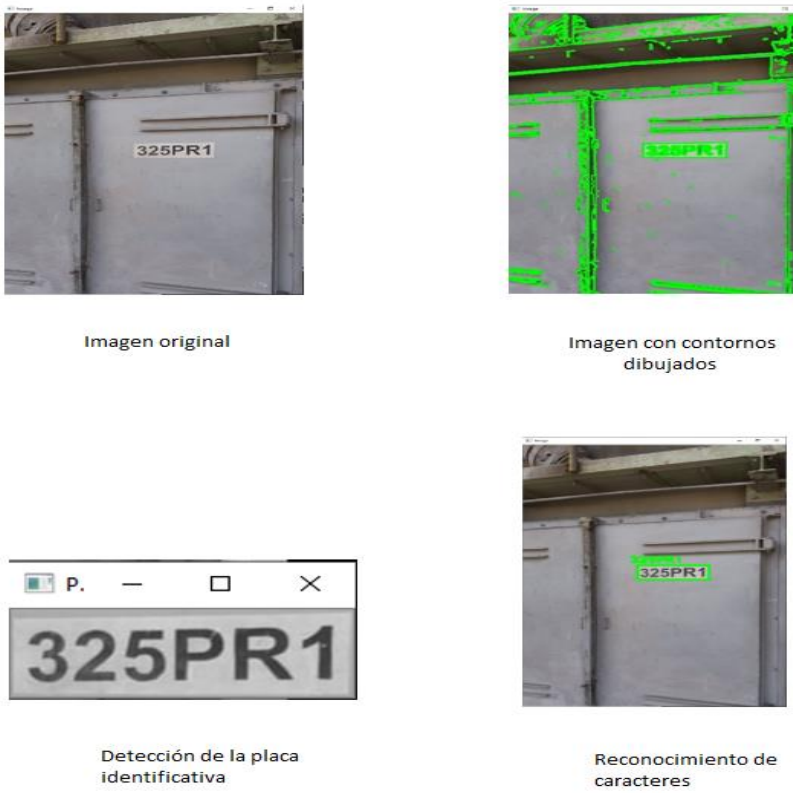
- Una vez reconocido la imagen se pasa por un filtro de caracteres especiales y minúsculas

En la siguiente figura 84 y 85 se observan los resultados en la placa identificativa en fondo uniforme y la placa identificativa de la fábrica.

Figura 84: Resultados del proceso de reconocimiento (335EB1)

Fuente: Elaboración Propia

Figura 85: Resultados del proceso de reconocimiento (325PR1)

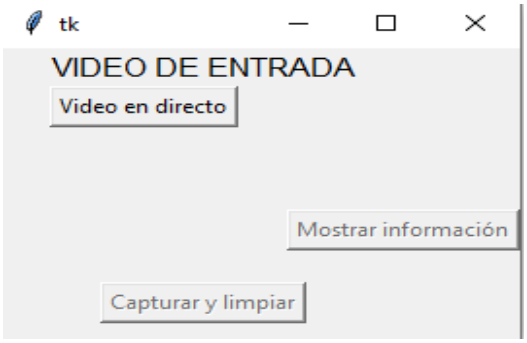


Fuente: Elaboración Propia

4.1.4 Resultados de la interfaz gráfica

Se obtiene una interfaz fácil de interpretar para el usuario, con un botón que permite ver el video en directo para procesar las imágenes en tiempo real (Video en directo), un botón que permite capturar la última imagen procesada y guardarla en una variable para que luego pueda ser mostrado en la interfaz, además de limpiar la interfaz con el botón (Capturar y limpiar), y por ultimo está el botón (Mostrar información) el cual muestra la información en tiempo real de la máquina identificada mostrando en que parte del proceso se encuentra, la placa eléctrica, mecánica, planes de mantenimiento, cronograma de mantenimiento y una breve descripción de la máquina. Se utiliza la librería Tkinter que permite realizar interfaces y configurar el boton correspondiente, además de la librería Image y Image Tk para poder mostrar las imágenes en la interfaz.

Figura 86: Interfaz del proyecto

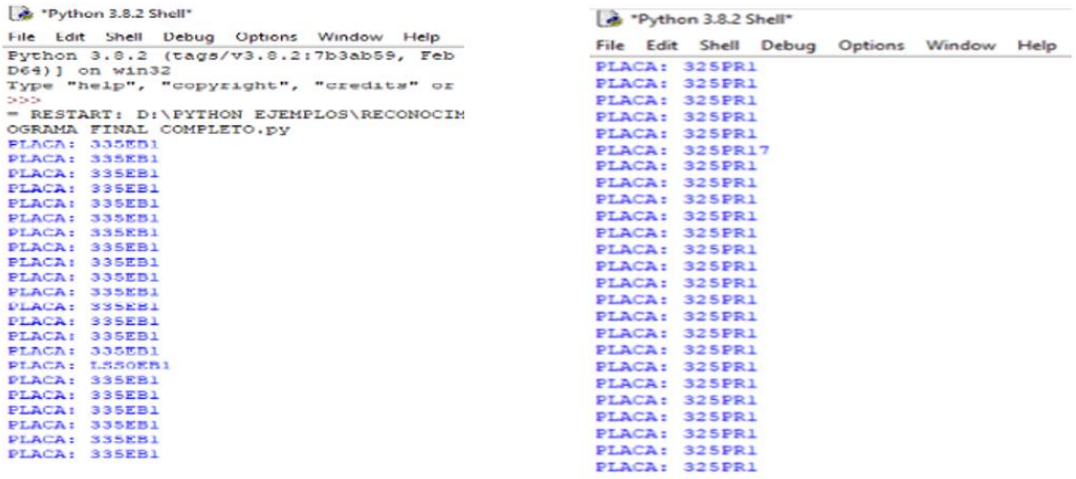


Fuente: Elaboración Propia

4.1.5 Resultados de la eficiencia del proyecto

Para obtener la eficiencia en el proyecto, se necesita saber el porcentaje de acierto que realiza el detector de caracteres para obtener el código correcto. Se va a tomar 100 imágenes en total de las 2 placas en fondo verde tomados de manera frontal a una distancia de 1.5 metros, y se analizará si el resultado obtiene un porcentaje de acierto mayor a 90%, la información es tomada igual como se muestra a continuación.

Figura 87: Resultados de las placas identificadas



Fuente: Elaboración Propia

En la siguiente tabla se pondrá la cantidad de imágenes exitosas y erróneas obtenidas del reconocimiento.

Tabla 18: Éxito y fallo en la detección de caracteres

N° de muestras	N° de Éxito	N° de Fallo
100	96	4

Fuente: Elaboración Propia

Se va a analizar la eficiencia con la probabilidad de éxito del proyecto, se utiliza la prueba de hipótesis de una proporción con cola derecha, usando una aproximación con nivel de confiabilidad de 90%.

Eficiencia: El proyecto es eficiente si el porcentaje de éxito al reconocer los caracteres de la placa es mayor al 90%.

Donde:

P: Probabilidad de éxito del reconocimiento de la placa del proyecto

Po: Probabilidad mínima que el proyecto debe de tener 90%

n: Numero de muestras

$\sigma_{\bar{P}}$: Desviación estándar de la distribución

Zc: Número de desviaciones estándar crítica

Zp: Número de desviaciones estándar de prueba

$\bar{P}$: Proporción muestral

Paso 1: Se presentan 2 ocasiones:

H0: La Probabilidad obtenida del proyecto es menor al 90% (P<Po)

H1: La Probabilidad obtenida del proyecto es mayor al 90% (P>Po)

Paso 2: Se obtiene de la tabla 1 del capítulo 2 que el valor Zc con cola derecha y confianza del 90% equivale a 1.28

$$\boldsymbol{Zc = Z_{1-\alpha} = Z_{0.90} = 1.28}$$

Paso 3: Se halla los valores críticos y de prueba

Vc: $\boldsymbol{Zc = 1.28}$

Vp: $Zp = \frac{\bar{P} - P_0}{\sigma_{\bar{P}}}$

Donde: $P_0 = 0.90$; $n = 100$

$$\boldsymbol{\bar{P} = \frac{Exito}{Muestras} = \frac{96}{100} = 0.96}$$

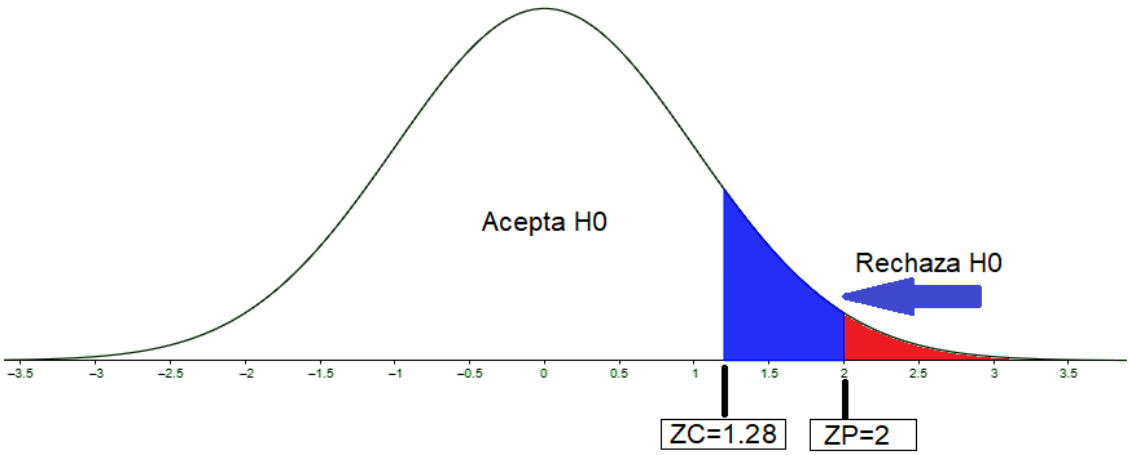
$$\sigma_{\bar{P}} = \sqrt{\frac{P_0 * (1 - P_0)}{n}} = 0.03$$

Por lo tanto:

$$Zp = \frac{0.96 - 0.90}{0.03} = 2$$

Paso 4: Establecer zona de aceptación/Rechazo de Ho como se observa en la figura 88

Figura 88: Resultados de las placas identificadas



Fuente: Elaboración Propia

Paso 5: Decisión y conclusión de la prueba:

Decisión:

Como $Z_p > Z_c$ y además $Z \in [1.28, +\infty]$ por lo tanto se rechaza H_0y se acepta H_1

Con la anterior se puede observar que la probabilidad de detección de la placa identificativa en el proyecto es mayor al 90% resultando que tiene suficiente confianza para que se apruebe la eficiencia del proyecto.

4.1.6 Resultados de la información deseada de la máquina reconocida

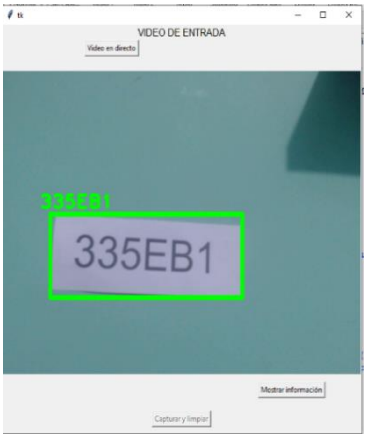
Le entrega de información luego de reconocer la placa identificativa es 100% eficiente. En la figura 89 se observa el resultado de la placa identificativa 325PR1 y en la figura 90 el resultado de la placa 335EB1 con su código correcto, además da la información deseada de la máquina que se encuentran en los datos almacenados cuando se presiona el botón de mostrar información.

Figura 89: Resultados de la información completa de la placa en fondo verde (325PR1)


```
PLACA: 325PR1
PLACA: 325PR1
PLACA: 325PR1
PLACA: 325PR1
La última placa capturada es: 325PR1
La placa identificada es: 325PR1
Se encuentra en los datos almacenados
PLACA IDENTIFICATIVA:
325PR1= Prensa de Rodillos, perteneciente al proceso de molienda de crudo,
nombrado como prensa de clinker 1
PLACA ELECTRICA:
Se encuentran en los 2 motores principales( 325PR1MT2 y 325PR1MT2)
PLACA MECANICA:
Se encuentra en la base de datos de Unacem
PLANES DE MANTENIMIENTO:
Se encuentra en la base de datos de Unacem
CRONOGRAMA DE MANTENIMIENTO:
Mantenimiento del Motor Principal (24/01/2021)
Mantenimiento del conducto (Obstrucción) (19/02/2021)
Mantenimiento del sector del eje (Observar fisuraas) (03/04/2021)
Ultimo mantenimiento el dia (03/04/2021)


DESCRIPCIÓN:
Encargado de moler el crudo, el material ingresa por el chute de entrada
para  que luego por 2 rodillos se tritura dando unaa gralumetria de
0.5mm, los rodillos estan impulsados por 2 grandes motores
(325PR1MT1 y 325PR1MT2) sujetados al piso, la prensa tiene un eje que
se mueve según la vibración, además tiene una cámara movil
encargado de empujar transversalmente a la cámara inmovil para
triturar el material.
```

Fuente: Elaboración Propia

Figura 90: Resultados de la información completa de la placa en fondo verde (335EB1)

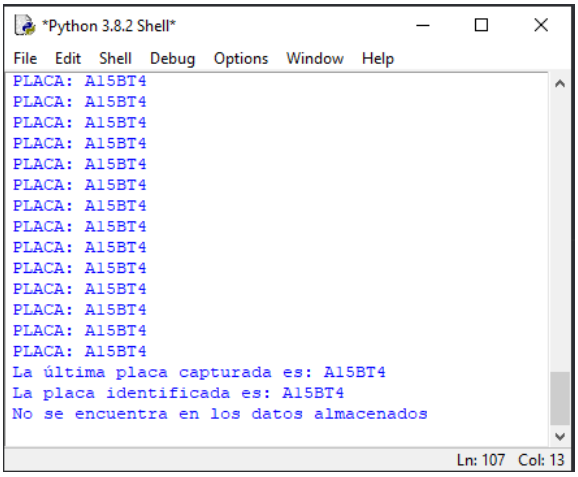



```
PLACA: 335EB1
PLACA: 335EB1
PLACA: 335EB1
PLACA: 335EB1
PLACA: 335EB1
La última placa capturada es: 335EB1
La placa identificada es: 335EB1
Se encuentra en los datos almacenados
PLACA IDENTIFICATIVA:
335EB1=Elevador de baldes, perteneciente al proceso de la molienda de crudo,en la zona de la prensa de clinker 1

PLACA ELECTRICA:
Se encuentran en el motor principal ( 335EB1MT1)
PLACA MECANICA:
Se encuentra en la base de datos de Unacem
PLANES DE MANTENIMIENTO:
Se encuentra en la base de datos de Unacem
CRONOGRAMA DE MANTENIMIENTO:
Mantenimiento de los baldes (Observar Rotura) (05/02/2021)
Mantenimiento de las fajas (Medir dureza) (24/03/2021)
Mantenimiento del motor (Vibración estable) (01/04/2021)
Ultimo mantenimiento el dia (01/04/2021)


DESCRIPCIÓN:
Encargado de transportar el material saliente del molino (325MO1) a
través de un motor que lleva el material por unos baldes en forma de
cuchara,  para posteriormente ser llevado a una compuerta electrica
que a través de una canaleta alimenta a la separadora de material
```

Fuente: Elaboración Propia

En la figura 91 se identifica un código inventado que no se encuentra en los datos almacenados (A15BT4), por lo tanto, se observa que nos da como resultado que el código detectado no se encuentra en la base de datos.

Figura 91: Resultados de la placa no encontrada en la base de datos (A15BT4)



Fuente: Elaboración Propia

4.2 Beneficios técnicos y Económicos

- **Corto plazo:** Ahorros en el tiempo operativo
- **Largo plazo:** Mejor rendimiento, costos óptimos y mejor utilidad.
- Rentabilidad en el proyecto debido que el valor obtenido en el análisis de Costo-Beneficio es mayor a 1
- Cumpliendo las consideraciones técnicas se eligen los equipos esenciales para el desarrollo, dando como resultado una alta eficiencia de 90%.
- La elección de las herramientas del proyecto es con cumplimiento de los requisitos mínimos y del menor costo.
- Con la tecnología elegida de visión artificial se puede implementar fácilmente en la empresa debido que utiliza como procesamiento a las placas de las máquinas instaladas
- Incremento de producción
- Llegada instantánea de información de las máquinas a los trabajadores cuando se encuentran en planta

4.3 Presupuesto

4.3.1 Herramientas

Tabla 19: Presupuesto de las herramientas

Herramientas	Costo (Soles)
Cámara Web Logitech C922 HD 720p	350 S/.
Ordenador AMD Ryzen 5 2600	2,500 S/.
Celular Galaxy J5(16Gb) SM-J500M	600 S/.
TOTAL	**3,450 S/.**

Fuente: Elaboración Propia

4.3.2 Mano de obra

Tabla 20: Presupuesto de la mano de obra

Actividad	Precio (Soles)
Desarrollo del sistema de reconocimiento de placas	2,000 S/.
Desarrollo del aplicativo Móvil	500 S/.
Desarrollo del Servidor	1,500 S/.
Implementación, Pruebas	500 S/.
Documentación del proyecto	1,200 S/.
TOTAL	**5,700 S/.**

Fuente: Elaboración Propia

4.3.3 Presupuesto total

Tabla 21: Presupuesto total

Concepto	Coste total (Soles)
Hardware	3,450 S/.
Mano de obra	5,700 S/.
Imprevistos	800 S/.
TOTAL	**9,950 S/.**

Fuente: Elaboración Propia

4.4 Análisis costo-Beneficio

Para desarrollar el análisis Costo-Beneficio se necesita saber la inversión, egresos e ingresos del proyecto, por lo cual en las siguientes tablas 2,3 y 4 se muestra de manera respectiva.

Tabla22: Inversión del proyecto

Detalles de inversión del desarrollo	Dinero en soles
Desarrollo	S/.4,000
Hardware	S/.3,450
Implementación, Pruebas	S/.500
Documentación	S/.1,200
Imprevistos	S/.800
TOTAL	S/.9,950

Fuente: Elaboración propia

La inversión que se realiza es de S/.9,950 equivalente al presupuesto total del proyecto.

Se realiza en la siguiente tabla3, el egreso mensual del proyecto y se analiza el gasto en 5 meses debido a los mantenimientos o actualizaciones de la base de datos.

Tabla23: Egresos del proyecto

Detalles de egreso	Mes 1	Mes 2	Mes 3	Mes 4	Mes 5
Mantenimiento de Celular	S/.100	0	S/.100	0	S/.100
Actualización de Base de datos	S/.250	0	S/.250	0	S/.250
Mantenimiento del Servidor Web	S/.500	0	S/.500	0	S/.500
Internet	S/.90	S/.90	S/.90	S/.90	S/.90
TOTAL	S/.940	S/.90	S/.940	S/.90	S/.940

Fuente: Elaboración propia

En la siguiente tabla4 se observa los ingresos que realiza el proyecto, en este caso es el ingreso mensual que se gana cuando el proyecto esté en funcionamiento, debido que las líneas de producción se activan lo más rápido posible.

Tabla24: Ingresos del proyecto

Detalles de ingresos	Mes 1	Mes 2	Mes 3	Mes 4	Mes 5
Reactivación de líneas de Proceso	S/.5,000	S/.7,000	S/.8,500	S/.4,000	S/.6,500
TOTAL	S/.27,400				

Fuente: Elaboración propia

Para obtener el costo - beneficio se necesita dividir los ingresos entre Costos-Inversión, el cual los datos se muestran en la siguiente tabla5, dando como resultado que el proyecto es rentable debido que el valor C/B es mayor a 1.

Tabla25: Obtención del Costo-Beneficio

Inversion	S/.9,950
Ingresos	S/.27,400
Egresos	S/.3,000
Costos-Inversión	S/.12,950
C/B	2.1158

Fuente: Elaboración propia

CONCLUSIONES

- El sistema de reconocimiento de la placa identificativa reconoce los caracteres con la librería Tesseract que con sus redes neuronales predefinidos y la configuración psm 11 busca la mayor cantidad de texto posible en la placa identificativa sin ningún orden en particular, se logra obtener caracteres de la placa, pero con algunos errores debido al ruido o al desenfoque que detectan caracteres que no se encuentran en la placa, para evitar ello se utiliza un filtro de caracteres especiales y minúsculas para obtener un porcentaje de éxito mayor al 90% como se mostró en el capítulo 3.

- Antes de realizar operaciones en una imagen se debe de observar que el histograma muestre un buen contraste e iluminación, además de saber cuál es la característica de la imagen que deseo extraer para realizar un buen reconocimiento.

- Se desarrolló una interfaz fácil de interpretar para el usuario con la librería Tkinter y ImageTk el cual son esenciales para configurar una interfaz y poder mostrar una imagen en ello, la interfaz contiene botones que tienen las funciones de mostrar el procesamiento en tiempo real, capturar la placa identificativa y mostrar la información de la máquina identificada.

- El sistema de reconocimiento reconoce el código de la placa e identifica si esta se encuentra en los datos almacenados, muestra únicamente la información de las máquinas que se encuentran en los archivos Excel, los códigos procesados que no se encuentren en los archivos son mostrado con el siguiente mensaje “Esa máquina no se encuentra en los datos almcenados”. El proyecto puede mejorarse creando

una base de datos externo con un lenguaje SQL para que se comunique con el programa y muestre la información necesaria.

- El procedimiento del reconocimiento es efectivo para los dos casos que se han presentado en el proyecto, ya sea para las placas identificativas con fondo verde y las placas identificativas de la fábrica, en el primer caso respectivo se demuestra que los procedimientos desde la adquisición de imagen hasta mostrar la información son óptimos para obtener la placa deseada, en el procedimiento del segundo caso se muestra que es óptimo cuando se obvia el método de la binarización pero igualmente existe dificultad si se quisiera tomar imágenes en tiempo real, debido a los diferentes ruidos y fondos que hace que el umbral varíe, por ello es recomendable tomar únicamente la placa con un fondo uniforme y realizar todo el procedimiento observado en el capítulo 3 del proyecto.

RECOMENDACIONES

- Para que las placas identificativas de la fábrica nos den siempre un porcentaje exitoso se recomienda tomar a 1 metro de distancia y poner a cada placa un fondo unicolor que diferencie a la placa y que se procese únicamente ese fondo con la placa para poder obtener el resultado con un éxito mayor al 90% como se realizó con las placas identificativas con fondo verde desarrollado en este Proyecto.

- Se recomienda que haya buena iluminación y pocos objetos en el fondo debido que el diseño del algoritmo depende de ello. Para discriminar la placa identificativa se utilizó un algoritmo que identifique los rectángulos de las áreas encerradas y que pertenezcan una cierta longitud y área.

- Se debe utilizar siempre las versiones mencionadas en el proyecto ya sea del programa principal Python V 3 o de las librerías porque estos al actualizarse cambian algunos datos y hace que la detección varíe.

BIBLIOGRAFIA

[1]. Castillo, J (2020). Diseño de un proceso de paletizado mediante procesamiento de imágenes manipulado por el robot industrial Kuka, en la empresa Abiexsa, Puente Piedra, año 2019. [Tesis de titulación, Universidad Tecnológica del Perú], Perú.

[2]. Herrera J. (2013). Diseño e implementación de una aplicación móvil basada en la tecnología nfc para acceso a información de las piezas de arte de un museo. [Tesis de titulación, Pontifica Universidad Católica del Perú], Perú

[3]. Delgado.L.(2010). Reconocimiento de placas vehiculares. [Tesis de maestría, Instituto politécnico Nacional], México

[4]. Peréz J (2014). Reconocimiento de placas vehiculares mediante procesamiento de imágenes para optimizar el acceso a los parqueaderos de la UTA, [Tesis de titulación, Universidad Técnica de Ambato], Ecuador

[5]. Vásquez y Melo (2018). Sistema automático de reconocimiento de placas vehiculares [Tesis de titulación, Universidad Cooperativa de Colombia], Colombia

[6]. Paredes y Guerrero (2012) Estudio comparativo entre algoritmos de reconocimiento de borde para identificación de placas de autos, [Tesis de titulación, Escuela superior politécnica de Chimborazo], Ecuador

[7]. Espinoza G. (2014) Sistema de reconocimiento de patrones en placas vehiculares para el acceso automático de visitas a un edificio, [Tesis de titulación, Pontifica Universidad Católica del Perú], Perú

[8]. Barreto y Lizarraga (2019). Modelo de Sistema de Reconocimiento Facial para el Control de la Trata de Personas, [Tesis de titulación, Universidad Tecnológica del Perú], Perú.

[9]. Diaz F. (2018) Investigación sobre variables predictivas del mantenimiento de parques eólicos, [Tesis de doctorado, Universidad de la Coruña], España

[10]. Guadalupe G. (2017) Reconocimiento de objetos utilizando OpenCV y Python en una Raspberry pi 2 en una tlapalería, [Tesis de titulación, Universidad autónoma del estado de México], México

[11]. Sailema F. (2017) Sistema electrónico de alerta automática para el reconocimiento de señales de tránsito reglamentarias, preventivas e informativas en la ciudad de Ambato, [Tesis de titulación, Universidad Técnica de Ambato], Ecuador

[12]. Rosales C. (2017) Prototipo de detección de expresiones corporales mediante visión artificial para mejorar la comunicación con niños que tienen parálisis cerebral infantil, [Tesis de titulación, Universidad Nacional de Loja], Ecuador

[13]. Garay, C (2019). Optimización de kpi's en la gestión de almacenes e integración con empresas del grupo UNACEM, [Tesis de maestría, Universidad Tecnológica del Perú], Perú.

[14]. Quispe, N (2016). Evaluación del sistema colector de polvo de la barredora tennant para disminuir sus emisiones en la empresa UNACEM, [Tesis de titulación, Universidad Nacional del centro del Perú], Perú.

[15]. Esteban, E (2017). Mantenimiento centrado en confiabilidad para el equipo más crítico del área de molienda de clinker en planta Atocongo – Lima, [Tesis de titulación, Universidad Nacional del centro del Perú], Perú.

[16]. Sobrado, E (2003) Sistema de Visión artificial para el reconocimiento y manipulación de objetos utilizando un brazo robot, [Tesis de maestría, Pontifica Universidad Católica del Perú], Perú

[17]. Viera, G (2017) Procesamiento de imágenes usando OpenCv aplicado en Raspberry Pi para la clasificación del cacao, [Tesis de titulación, Universidad de Piura], Perú

[18]. Nieto, A (2018) Evolución del procesamiento digital de imágenes, [Tesis de titulación, Universidad autónoma del estado de México], México

[19]. Boris, R (2006) Procesamiento digital de imágenes. Recuperado de (http://lapi.fi-p.unam.mx/wp-content/uploads/PDI_Cap2_Fundamentos_de_la_Imagen_Digital.pdf)

[20]. López, J (2009) Análisis de imágenes microscópicas para la determinación de la cantidad y el tamaño de larvas de concha de abanico, [Tesis de titulación, Pontifica Universidad Católica del Perú], Perú

[21]. Vicente, V (2015) El histograma de una imagen digital. Escuela Técnica Superior de Ingeniería Informática, faculta de ingeniería, España. Recuperado de (https://riunet.upv.es/bitstream/handle/10251/12711/El_histograma_una_imagen_digital.pdf?sequence=1)

[22]. Triana,N., Jaramillo, A., Gutierrez R,. Cesar R (2016) Técnicas de umbralización para el procesamiento digital de imágenes de GEMFoils, Volumen (21), 352-359

[23]. Alvaro, Carlos (2007) Técnicas de procesamiento de imágenes bidimensionales, [Tesis de titulación, Universidad tecnológica de Bolívar], Colombia

[24]. R.C. González, R.E. Woods (2008) Digital Image Processing, Pearson Prentice Hall.

[25]. Ruíz, F (2010) La transformada de Fourier Aplicación al filtrado de imágenes. Universidad politécnica de Valencia, España.

[26]. García, E (2008) Detección y clasificación de objetos dentro de un salón de clases empleando técnicas de procesamiento digital de imágenes, [Tesis de maestría, Universidad autónoma metropolitana], México.

[27]. García, P (2013) Reconocimiento de imágenes utilizando redes neuronales artificiales, [Tesis de maestría, Universidad complutense de Madrid], España.

[28]. Arias, J (2017) Plataforma de evaluación de algoritmos de reconocimiento de caracteres numéricos en imágenes digitales, [Tesis de maestría, Escuela superior politécnica de Chimborazo], Ecuador.

[29]. Collaguazo, J (2018) Sistema electrónico para facilitar la comunicación y actividades cotidianas de los enfermos con esclerosis lateral amiotrófica (ELA), [Tesis de titulación, Universidad Técnica de Ambato], Ecuador

[30]. Vega H., Cortez A., Huayna A. (2009) Reconocimiento de patrones mediante redes neuronales artificiales, Universidad Mayor de San Marcos, Perú

ANEXOS

[ANEXO A.1]

Programación de la detección en tiempo real de placas identificativas en fondo verde

PROGRAMA FINAL COMPLETO.py - D:\PYTHON EJEMPLOS\RECONOCIMIENTO DE PLACAS\PRGRAMA FINAL COMPLETO\

File Edit Format Run Options Window Help

```
from tkinter import *
from tkinter import filedialog
from PIL import Image
from PIL import ImageTk
import pytesseract
import cv2
import xlrd
import re
pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-OCR\tesseract'
Excel1= r'D:\PYTHON EJEMPLOS\RECONOCIMIENTO DE PLACAS\Prensa325PR1.xlsx'
Excel2= r'D:\PYTHON EJEMPLOS\RECONOCIMIENTO DE PLACAS\Placa335EB1.xlsx'
def Mostrar():
    global cap
    cap.release()
    img = ImageTk.PhotoImage(image=im)
    Video.configure(image=img)
    Video.image = img
    print('La placa identificada es:',textoR)

    if textoR=='325PR1':
        print('Se encuentra en los datos almacenados')
        AbrirExcel= xlrd.open_workbook(Excel1)
        Hoja=AbrirExcel.sheet_by_name("Hoja1")

        for i in range(Hoja.nrows):
            print(Hoja.cell_value(i,0))

    elif textoR=='335EB1':
            print('Se encuentra en los datos almacenados')
            AbrirExcel= xlrd.open_workbook(Excel2)
            Hoja=AbrirExcel.sheet_by_name("Hoja1")

            for i in range(Hoja.nrows):
                print(Hoja.cell_value(i,0))
    else:
        print('No se encuentra en los datos almacenados')

def video_de_entrada():
    global cap

    Boton3.configure(state="active")
    Boton2.configure(state="disabled")
    Boton1.configure(state="active")
    InfVideo.configure(text="")
    cap = cv2.VideoCapture(0, cv2.CAP_DSHOW)
    visualizar()
```

```
  def visualizar():
      global cap
      global im
      ret, frame = cap.read()
      if ret == True:
          frame = Deteccion_placa(frame)
          frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
          im = Image.fromarray(frame)
          img = ImageTk.PhotoImage(image=im)
          Video.configure(image=img)
          Video.image = img
          Video.after(10, visualizar)


      else:
          Video.image = ""
          InfVideo.configure(text="")
          Boton1.configure(state="active")
          selected.set(0)
          Boton3.configure(state="disabled")
          cap.release()
```

```
def Deteccion_placa(frame):
    global text
    global textoR
    Patron= '[a-z]'
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    gray = cv2.blur(gray,(3,3))
    _,binarizada=cv2.threshold(gray,155,255,cv2.THRESH_BINARY)
    canny = cv2.Canny(binarizada,100,150)
    canny = cv2.dilate(canny,None,iterations=1)
    Contornos,_ = cv2.findContours(canny,cv2.RETR_LIST,cv2.CHAIN_APPROX_SIMPLE)
    for c in Contornos:
      Area = cv2.contourArea(c)
      X,Y,W,H = cv2.boundingRect(c)
      Epsilon = 0.09*cv2.arcLength(c,True)
      AproxRect = cv2.approxPolyDP(c,Epsilon,True)
      if len(AproxRect)==4 and Area>10000:
        Aspecto = float(W)/H

        if Aspecto>2.4:
          Placa= gray[Y:Y+H,X:X+W]
          Configuracion= r'--psm 11'
          text= pytesseract.image_to_string(Placa,config=Configuracion)

          if not text=='':
              textoR=text
              textoR = ''.join(filter(str.isalnum, textoR))
              textoR =re.sub(Patron,'', textoR)
              print('PLACA:',textoR)
              frame= cv2.rectangle(frame,(X,Y),(X+W,Y+H),(0,255,0),3)
              cv2.putText(frame,textoR,(X-20,Y-10),1,2.2,(0,255,0),3,cv2.LINE_AA)
          else:
              frame= cv2.rectangle(frame,(X,Y),(X+W,Y+H),(0,255,0),3)
              cv2.putText(frame,text,(X-20,Y-10),1,2.2,(0,255,0),3,cv2.LINE_AA)
    return frame


def Capturar_limpiar():
    cap.release()
    Video.image = ""
    InfVideo.configure(text="")
    Boton2.configure(state="active")
    Boton1.configure(state="active")
    selected.set(0)
    print('La última placa capturada es:',textoR)

cap = None
root = Tk()

Titulo = Label(root, text="VIDEO DE ENTRADA", font="bold")
Titulo.grid(column=0, row=0, columnspan=2)

selected = IntVar()

Boton1 = Button(root, text="Video en directo", state="active", command=video_de_entrada)

Boton1.grid(column=0, row=1)

InfVideo = Label(root, text="", width=20)
InfVideo.grid(column=0, row=2)
Video = Label(root)
Video.grid(column=0, row=3, columnspan=2)
Boton2 = Button(root, text="Mostrar información", state="disabled", command=Mostrar)
Boton2.grid(column=1, row=4, columnspan=2, pady=10)
Boton3 = Button(root, text="Capturar y limpiar", state="disabled", command=Capturar_limpiar)
Boton3.grid(column=0, row=5, columnspan=2, pady=10)
root.mainloop()
```

[ANEXO A.2]

Programación de la detección de imágenes de placas identificativas de la fábrica

*PROGRAMA IMAGENES SIN BINARIZAR.py - D:\PYTHON EJEMPLOS\RECONOCIMIENTO DE PLACAS\PROGRAMA COMPLETO IM.

File Edit Format Run Options Window Help

```
import cv2
import pytesseract
pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-OCR\tesseract'
Placa = []
image = cv2.imread('Prensa001.jpg')
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
gray = cv2.blur(gray,(2,2))
canny = cv2.Canny(gray,70,80)
canny = cv2.dilate(canny,None,iterations=1)
cnts,_ = cv2.findContours(canny,cv2.RETR_LIST,cv2.CHAIN_APPROX_SIMPLE)

for c in cnts:
  Area = cv2.contourArea(c)
  X,Y,W,H = cv2.boundingRect(c)
  Epsilon = 0.09*cv2.arcLength(c,True)
  ApproxRect = cv2.approxPolyDP(c,Epsilon,True)

  if len(ApproxRect)==4 and Area>2000:

    Aspecto= float(W)/H
    if Aspecto>2.4:
       Placa= gray[Y:Y+H,X:X+W]
       Configuracion= r'--psm 11'
       text= pytesseract.image_to_string(Placa,config=Configuracion)

       cv2.rectangle(image,(X,Y),(X+W,Y+H),(0,255,0),3)
       cv2.putText(image,text,(X-20,Y-10),1,2.2,(0,255,0),3)

cv2.imshow('Image',image)
cv2.moveWindow('Image',45,10)
cv2.waitKey(0)
```

[ANEXO A.3]

Programación del Histograma en grises

Prueba Histo Gray.py - D:\PYTHON EJEMPLOS\RECONOCIMIENTO DE PLACAS\HISTOGRAMA.

File Edit Format Run Options Window Help

```
import cv2
from matplotlib import pyplot as plt
Imagen=cv2.imread('IMAGEN.jpg')
Gray=cv2.cvtColor(Imagen, cv2.COLOR_RGB2GRAY)
Histo=cv2.calcHist([Gray],[0],None,[256],[0,255])
Figura=plt.figure(1)
Plot1=Figura.add_subplot()
Plot1.plot(Histo)
plt.title('Histograma')
Figura.show()
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Programación del Histograma en BGR

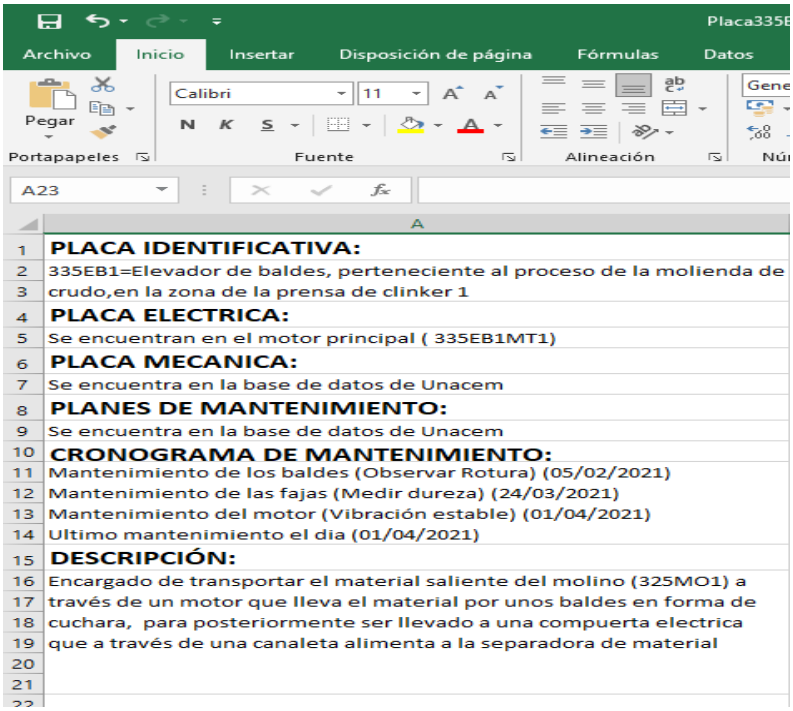
HISTO RGB.py - D:/PYTHON EJEMPLOS/RECONOCIMIENTO DE PLACAS/HISTOGRAMAS Y CA...

File Edit Format Run Options Window Help

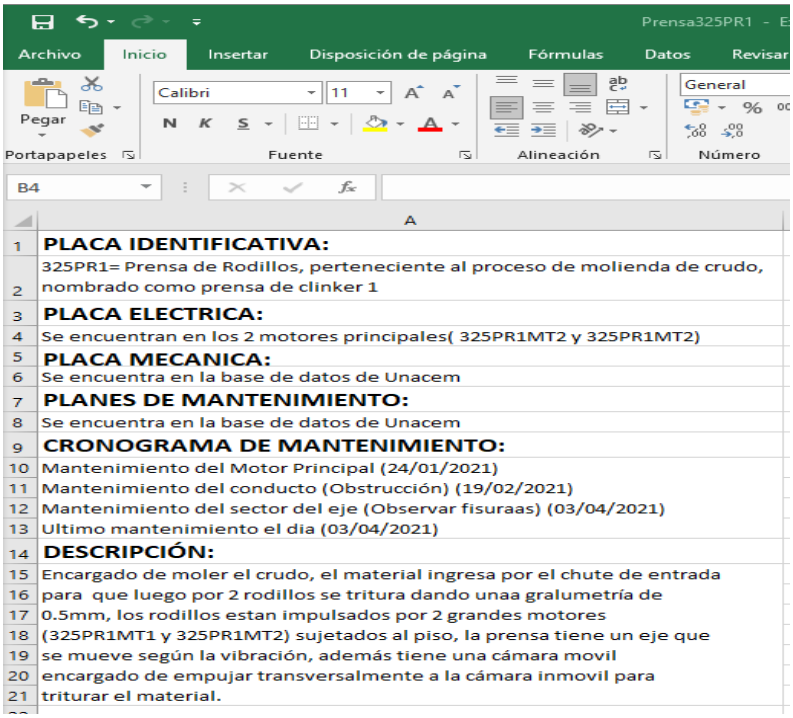
```
import cv2
from matplotlib import pyplot as plt
Imagen=cv2.imread('IMAGEN.jpg')
Colores=('b','g','r')
for i,col in enumerate(Colores):
    Histograma=cv2.calcHist([Imagen],[i],None,[256],[0,256])
    plt.plot(Histograma,color= col)
    plt.xlim([0,256])
plt.show()
```

[ANEXO A.4]

Datos almacenados de las placas 335EB1 y 325PR1 en Excel



	A
1	**PLACA IDENTIFICATIVA:**
2	335EB1=Elevador de baldes, perteneciente al proceso de la molienda de
3	crudo,en la zona de la prensa de clinker 1
4	**PLACA ELECTRICA:**
5	Se encuentran en el motor principal (335EB1MT1)
6	**PLACA MECANICA:**
7	Se encuentra en la base de datos de Unacem
8	**PLANES DE MANTENIMIENTO:**
9	Se encuentra en la base de datos de Unacem
10	**CRONOGRAMA DE MANTENIMIENTO:**
11	Mantenimiento de los baldes (Observar Rotura) (05/02/2021)
12	Mantenimiento de las fajas (Medir dureza) (24/03/2021)
13	Mantenimiento del motor (Vibración estable) (01/04/2021)
14	Ultimo mantenimiento el dia (01/04/2021)
15	**DESCRIPCIÓN:**
16	Encargado de transportar el material saliente del molino (325MO1) a
17	través de un motor que lleva el material por unos baldes en forma de
18	cuchara, para posteriormente ser llevado a una compuerta electrica
19	que a través de una canaleta alimenta a la separadora de material
20	
21	



	A
1	**PLACA IDENTIFICATIVA:**
2	325PR1= Prensa de Rodillos, perteneciente al proceso de molienda de crudo, nombrado como prensa de clinker 1
3	**PLACA ELECTRICA:**
4	Se encuentran en los 2 motores principales(325PR1MT2 y 325PR1MT2)
5	**PLACA MECANICA:**
6	Se encuentra en la base de datos de Unacem
7	**PLANES DE MANTENIMIENTO:**
8	Se encuentra en la base de datos de Unacem
9	**CRONOGRAMA DE MANTENIMIENTO:**
10	Mantenimiento del Motor Principal (24/01/2021)
11	Mantenimiento del conducto (Obstrucción) (19/02/2021)
12	Mantenimiento del sector del eje (Observar fisuraas) (03/04/2021)
13	Ultimo mantenimiento el dia (03/04/2021)
14	**DESCRIPCIÓN:**
15	Encargado de moler el crudo, el material ingresa por el chute de entrada
16	para que luego por 2 rodillos se tritura dando unaa gralumetría de
17	0.5mm, los rodillos estan impulsados por 2 grandes motores
18	(325PR1MT1 y 325PR1MT2) sujetados al piso, la prensa tiene un eje que
19	se mueve según la vibración, además tiene una cámara movil
20	encargado de empujar transversalmente a la cámara inmovil para
21	triturar el material.